**Лабораторна робота №**

**Інтерфейс гри (UI). Адаптивне меню**

**Мета**: ознайомитися з базовими елементами Unity User Interface. Навчитися створювати адаптивне меню.

**Література**

1. ….
2. …..

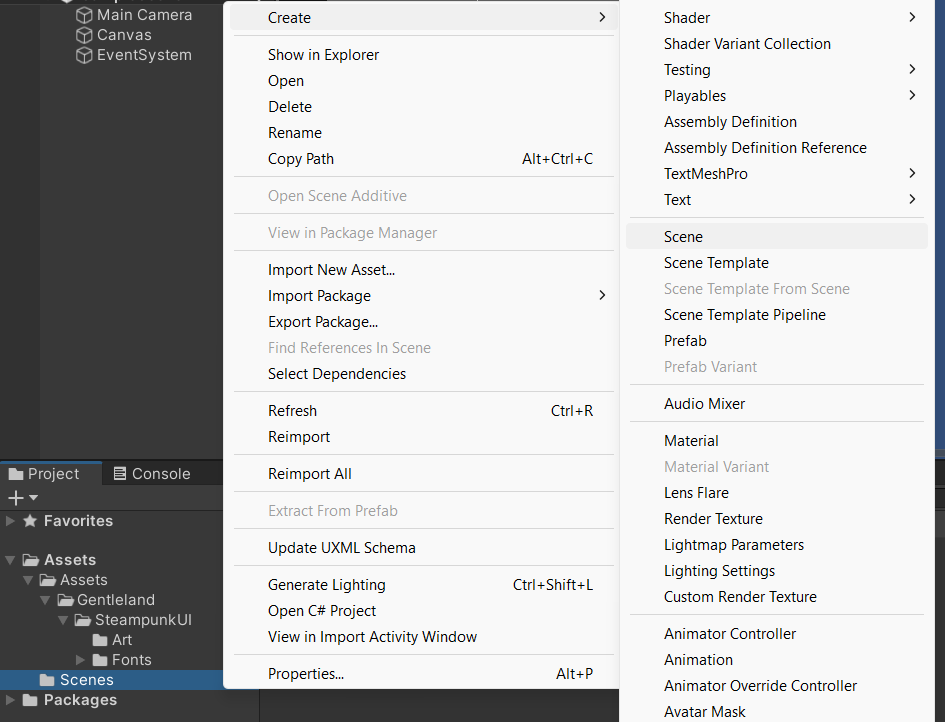
**Зміст роботи**

**Завдання 1.** *Створити меню до 2D гри, використовуючи для цього елементи управління GUI.*

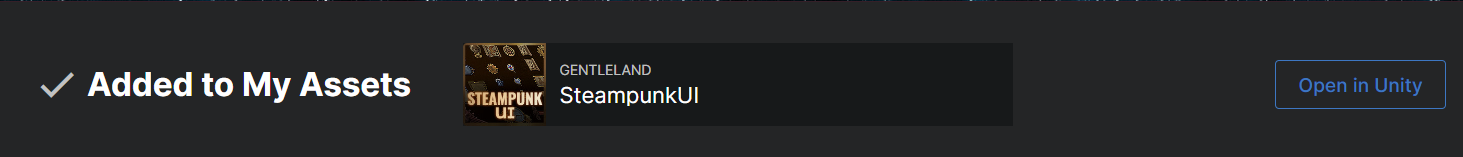
**Методичні рекомендації**

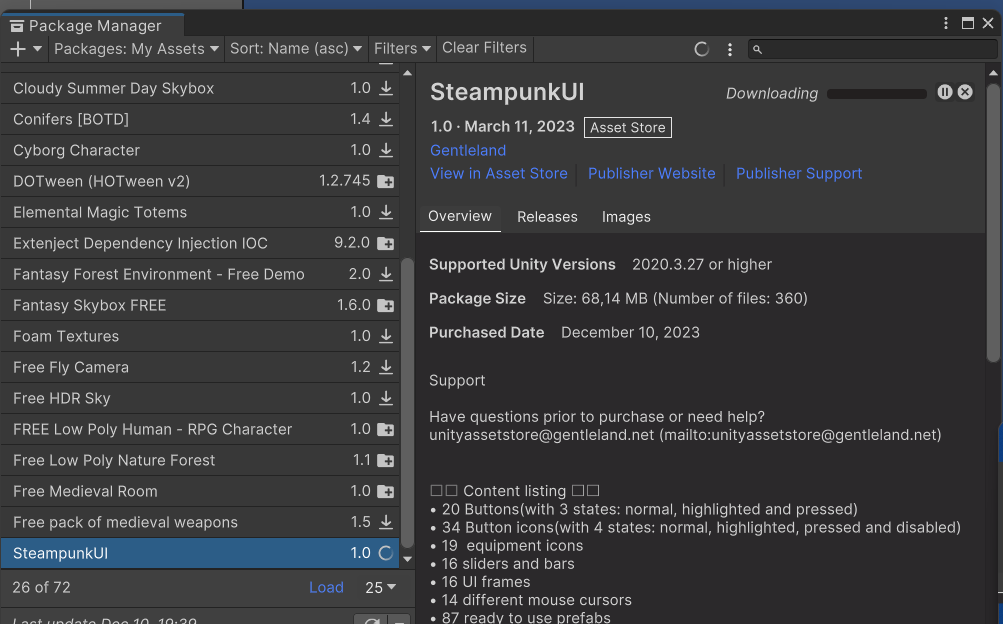
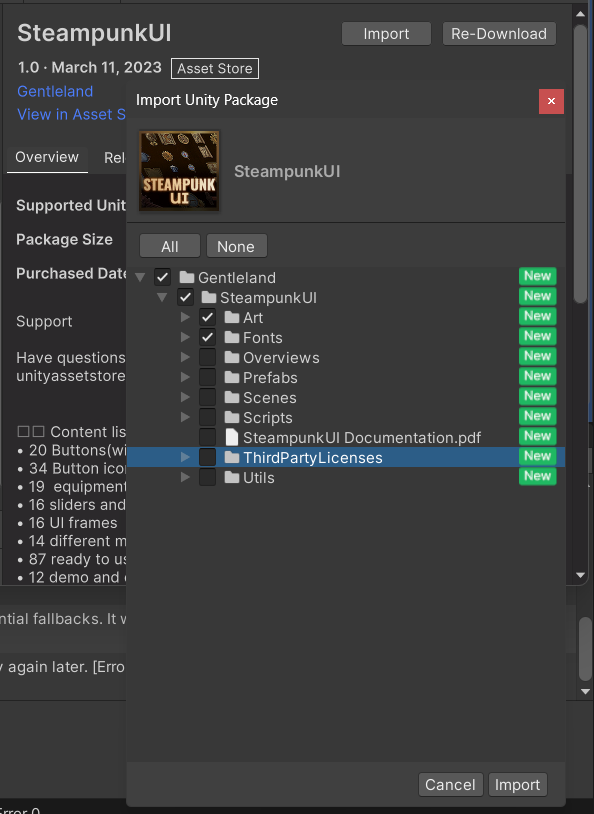
**…….**

1. Перед тим як перейти до створенню меню, необхідно зробити певні кроки.
   1. Створюємо проект , якщо його ще не має. Якщо проект вже є, тоді потрібно створити нову сцену для меню.

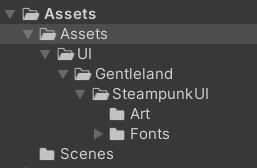


* 1. Скачуємо ассети для роботи над меню з unity assets store: <https://assetstore.unity.com/packages/2d/gui/icons/steampunkui-238976>   
     Ассети можна вибрати на свій розсуд, але не можна порушувати авторські права.
  2. Нажимаємо кнопку “Open in Unity”. У відкритому Package Manager нажимаємо кнопку “Download” навпроти назви ассету. Чекаємо завершення завантаження. Після того як завершилось завантаження нажимаємо кнопку “Import”. Ставимо галочки навпроти папок, які будуть імпортуватись в проект.

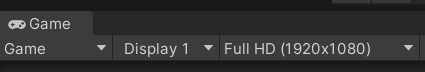


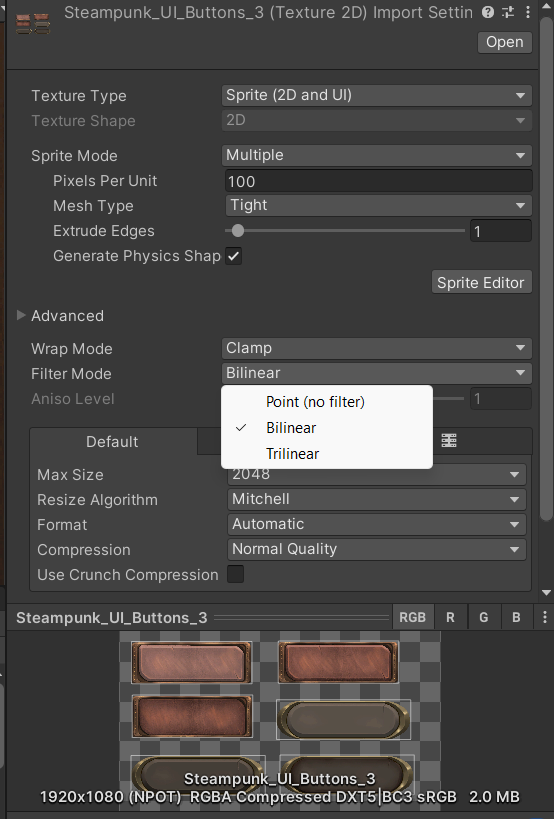
* 1. Створюємо папку в корні проекту з назвою Assets. Надалі всі ассети, які стосуються графіки, будуть знаходитись в цій папці. Далі створюємо папку UI та переносемо завантажені ассети в цю папку.



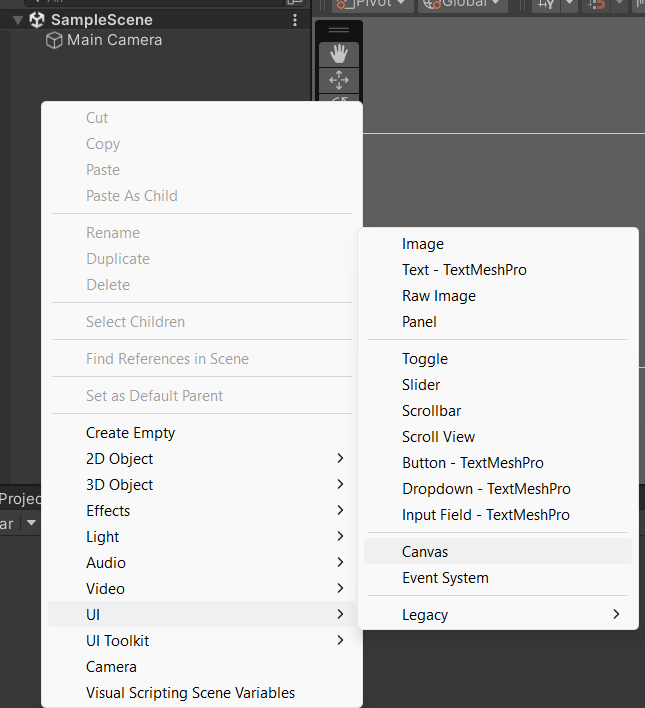
* 1. Замініть Free Aspect Ratio на 1920х1080



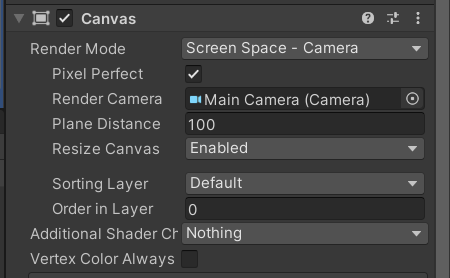
* 1. Для того щоб спрайти не виглядали розмито можна змінити налаштування Filter Mode. Filter Mode визначає метод, який використовується для семплювання текстур при їх зменшенні або збільшенні. Це впливає на якість і вигляд текстур в грі. Існують три основних режими фільтрації:
* Point - використовується для зменшення чи збільшення текстури без будь-якого згладжування. Це найбільш жорсткий режим, і він може призводити до артефактів або погіршення якості при збільшенні.
* Bilinear - використовується для семплювання текстур із згладжуванням лінійного типу. Він надає кращу якість, особливо при збільшенні текстур, порівняно з режимом Point, але витрачає більше ресурсів.
* Trilinear - це розширення Bilinear фільтра з додатковим міжмасштабуванням для усунення артефактів, пов'язаних із зміною рівня деталізації між різними рівнями міпмап. Використовується для текстур, які мають міпмапи (різні версії текстури для різних рівнів деталізації).



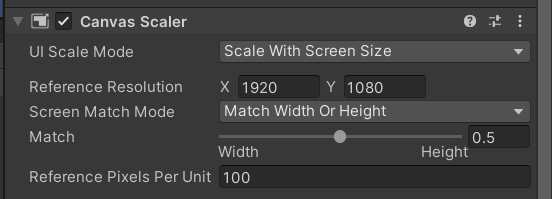
1. Створюємо Canvas в ієрархії об’єктів сцени. Нажимаємо правою клавішою мишки та додаємо об’єкт Canvas до сцени.

****

* 1. Налаштовуємо Render Mode компоненти Canvas (Його можна знайти в інспекторі об’єкта Canvas)
* У режимі ***Screen Space Overlay*** - канвас рендериться над усіма іншими об'єктами, без врахування їхньої глибини. UI розміщується в оверлейному просторі екрану. Цей режим часто використовується для простих UI-елементів, які повинні бути завжди видимими на екрані, незалежно від того, що відбувається у тривимірному просторі.
* У режимі ***Screen Space Camera*** - канвас рендериться у просторі камери, враховуючи перспективу та глибину сцени. Канвас може взаємодіяти з тривимірними об'єктами та розміщуватися у тривимірному просторі. Використовується, коли UI повинен взаємодіяти з тривимірним світом, наприклад, коли ви хочете вбудувати UI в тривимірну сцену.
* ***Pixel Perfect*** у компоненті Canvas є налаштуванням, яке забезпечує, що ваш UI буде відображатися без розмиття або стискання пікселів навіть на різних роздільностях екрану. Це корисно, особливо коли ви працюєте з піксель-артом або хочете забезпечити чітке відображення UI елементів. Якщо ви використовуєте TextMeshPro для текстових елементів, Pixel Perfect допомагає досягти чіткого відображення шрифтів. *Важливо враховувати, що Pixel Perfect ефективно працює тільки при використанні орієнтовної роздільності (reference resolution) в компоненті Canvas. Reference resolution - це роздільність, яку ви вказуєте для розробки, і Pixel Perfect працює на основі цієї роздільності, вирівнюючи об'єкти на Canvas відповідно.*

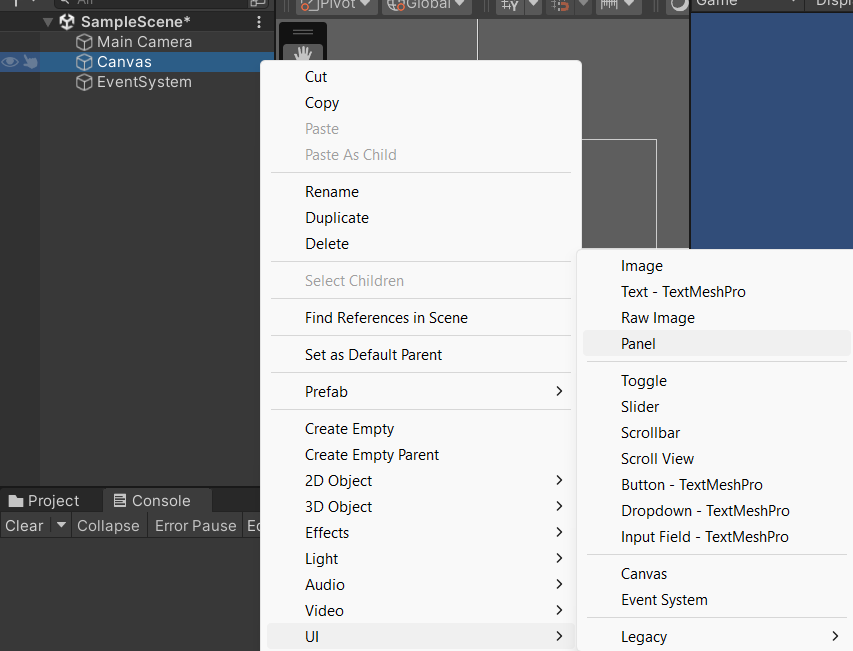


* 1. Налаштовуємо Canvas Scaler (Його можна знайти в інспекторі об’єкта Canvas)
* Canvas Scaler в Unity використовується для налаштування масштабування та розташування UI елементів, щоб забезпечити адаптивність на різних роздільностях екрану.
* Переконайтеся, що Scale Mode встановлено ​​на "Scale With Screen Size". Цей режим дозволить маштабувати Canvas в залежності від розміру екрану пристроя.
* Встановіть Reference Resolution у значення 1920х1080.
* Встановіть Screen Match Mode у значення Match Width or Height , що дозволить змінювати розміри канвасу в залежності від ширини та висоти екрану.
* Параметр Match визначає, чи використовується при масштабуванні ширина або висота як еталон, або їх поєднання. Встановіть його в значення 0.5, якщо ви хочете щоб Canvas масштабувався рівномірно по висоті і по ширині екрану.
* Reference Pixels Per Unit вказує на те, скільки пікселів в одиниці довжини. Це значення впливає на розміри та масштаб елементів Canvas. Збільшення розмірів може бути корисним, коли вам потрібно більше деталей або коли гра орієнтована на пристрої з високою роздільною здатністю.

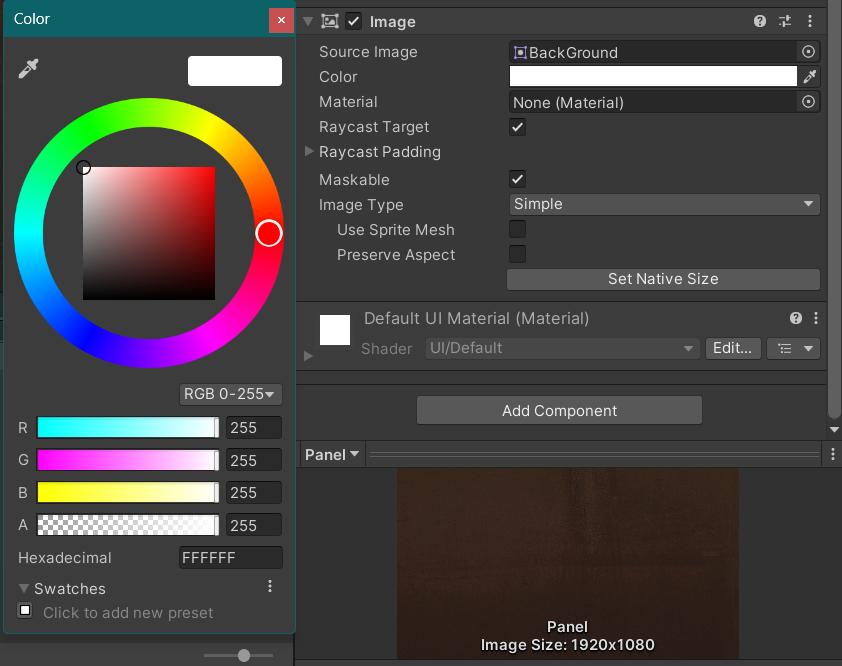


1. Створюємо фон для нашого меню. Для того щоб додати до Canvas новий об’єкт, нажимаємо на Canvas правою клавішою миші та обираємо Panel.

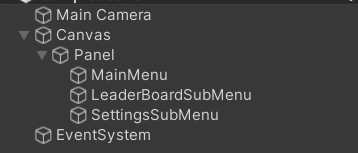
* Panel (панель) використовується для групування і розміщення інших елементів UI, таких як текст, зображення або інші панелі. Він є контейнером, який дозволяє організувати та структурувати елементи інтерфейсу на сцені.
* Деякі ефекти, такі як затемнення або розмиття, можуть бути легше застосувати до всього елемента Panel, ніж до кожного окремого дочірнього елемента.

****

* 1. На об’єкті Panel в інспекторі знаходимо компонент Image і міняємо його параметр Source Image. Далі модифікуємо параметр Color, а саме його властивість A(Alpha відповідає за прозорість) ставимо у значення 255.

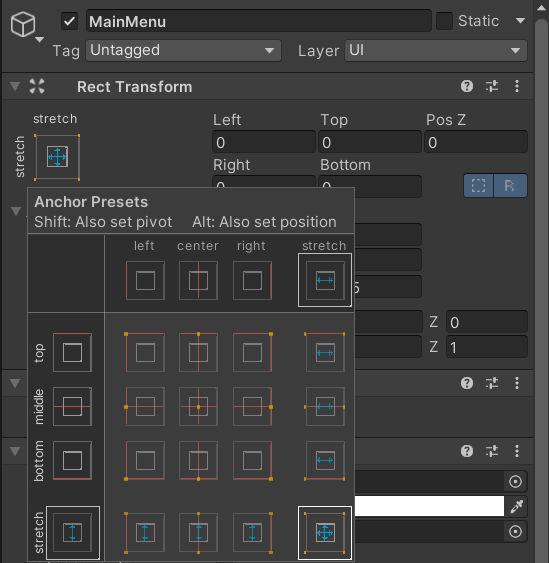
****

1. Додаємо до об’єкту Canvas ще панель, в якій буде знаходитись головне меню. З головного меню можна буде перейти в підменю. Для кожного підменю створяється на канвасі панель та називається відповідно. Кількість підменю залежить від функціоналу вашого меню. *Коли ви додасте та налаштуєте головне меню ви можете просто його продублювати (нажати на об’єкт меню лівою клавішою миші і використати комбінацію клавіш Ctrl + D ), таким чином прискоривши процес розробки.*



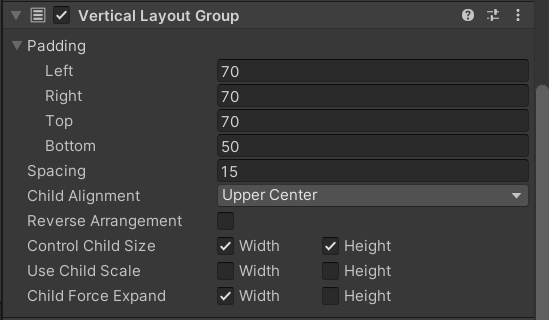
1. Налаштовуємо об’єкт MainMenu, який буде контейнером головного меню. Оберіть Source Image компонента Image, який вам подобається. Перейдіть в Inspector та за допомогою Anchor Preset встановіть розтяг об’єкта по всьому екрану.

* Anchor Presets визначають якорівні точки (anchors) - це чотири точки (верхня ліва, верхня права, нижня ліва, нижня права), які визначають положення та розмір елемента. Пресети надають зручний спосіб вибору конфігурацій якорів за замовчуванням для різних ситуацій. Наприклад, якщо ви обираєте пресет "Stretch", елемент буде розтягуватися по всій доступній ширині і висоті вказаної області. Якщо ви обираєте пресет "Top Left", елемент залишиться прикріпленим до верхнього лівого краю області.



1. Додайте до MainMenu панель та назвіть його ButtonsGroup. Додайте Source Image до компонента Image. До ButtonsGroup додайте компонент Vertical Layout Group. Далі налаштуємо цей компонент. Vertical Layout використовується для автоматичного розміщення елементів в вертикальному порядку.

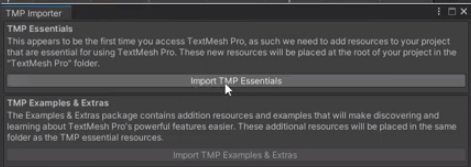
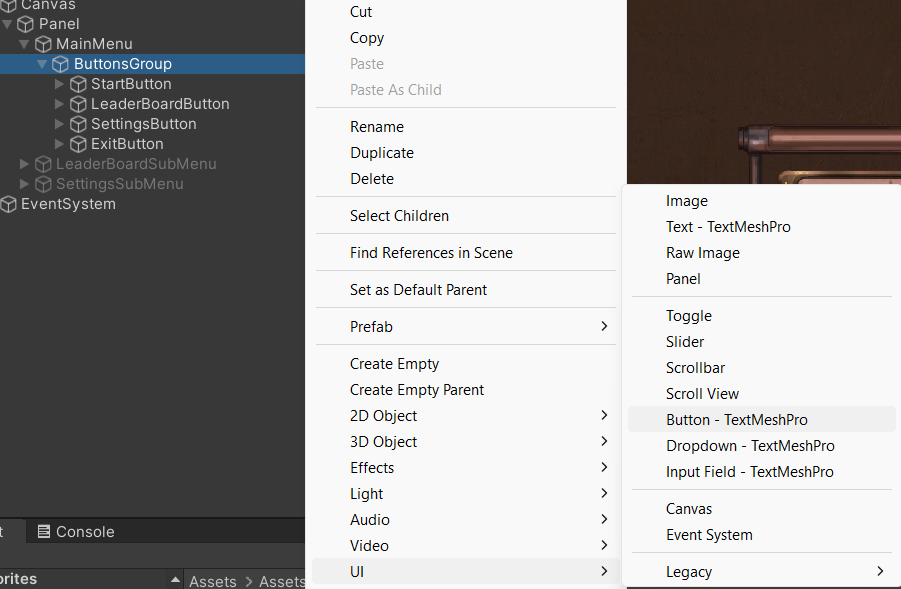
* **Padding** (Відступи) - визначає відстань між дочірніми елементами та межами Vertical Layout Group. Ви можете налаштовувати відступи з усіх сторін (верх, низ, ліво, право) незалежно.
* **Spacing** (Відстань) - визначає відстань між сусідніми дочірніми елементами у вертикальному списку.
* **Child Alignment** (Вирівнювання дочірніх елементів) - визначає, як дочірні елементи вирівнюються у вертикальному просторі. Ви можете вибрати між вертикальним центруванням, вгорі, внизу тощо.
* **Control Child Size** (Контроль ширини/висоти дочірніх елементів) - Ці параметри дозволяють вам визначити, як компонент контролює ширину або висоту дочірніх елементів.
* **Use Child Scale Width/Height** (Масштабування ширини/висоти дочірніх елементів) - розмір дочірніх елементів може масштабуватися відповідно.
* **Child Force Expand** (Примусове розширення дочірніх елементів) - вказує, чи потрібно дочірнім елементам розширюватися на весь розмір Vertical Layout Group.



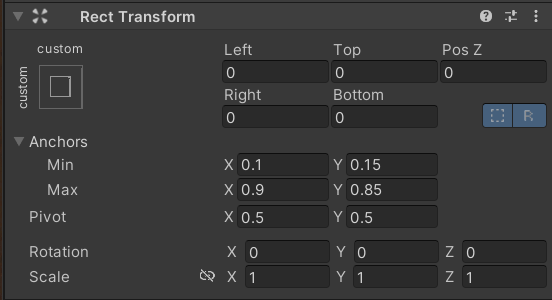
* 1. Anchors використовуються для прив'язки елементів інтерфейсу до різних країв або середини батьківського елемента. Anchors визначають, як розташовується та масштабується елемент при зміні розміру батьківського елемента або контейнера. Задайте мінімальні та максимальні значення якорів. Виставте параметри Left,Top,Right,Bottom у значення 0.



1. Додайте до об’єкта ButtonsGroup одну кнопку. Імпортуйте TextMeshPro

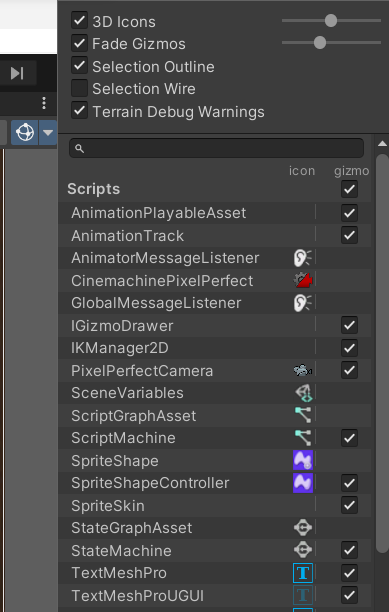


* 1. Далі потрібно налаштувати якорі для текста кнопки. Оскільки кнопка має картинку з рамочкою, потрібно розташувати її текст так щоб він не перекрив цю рамочку.

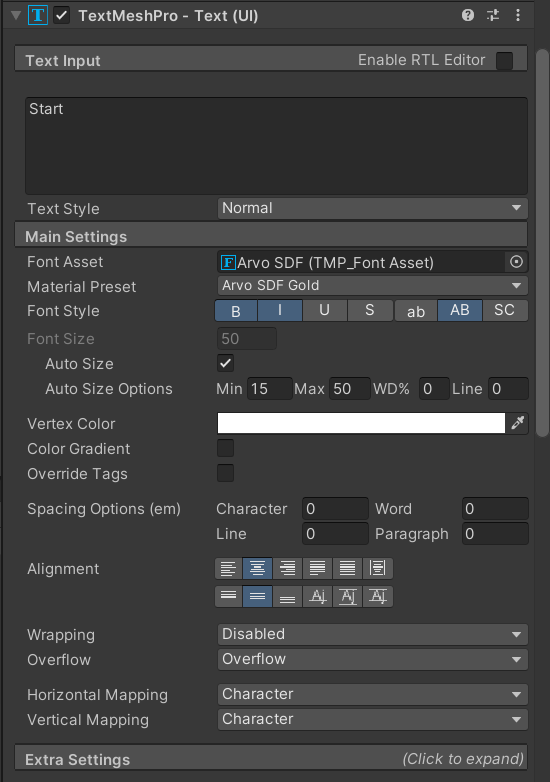




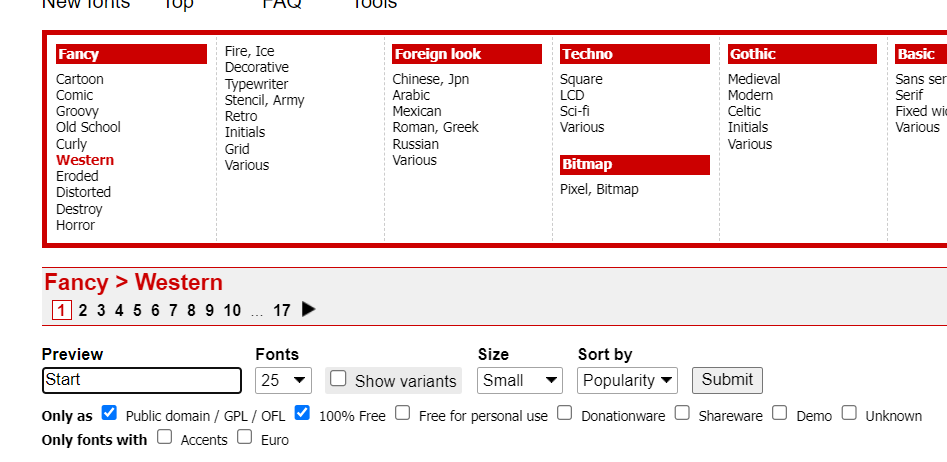
* 1. *(Опціонально) Якщо вам заважає іконка TextMeshPro, яка відображається в сцені, ви її можете прибрати у вкладці Gizmos. Нажміть на іконку з літерою Т навпроти TextMeshProUGUI.*



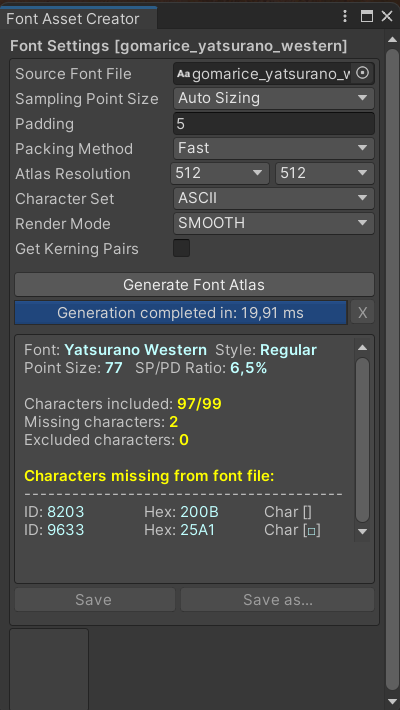
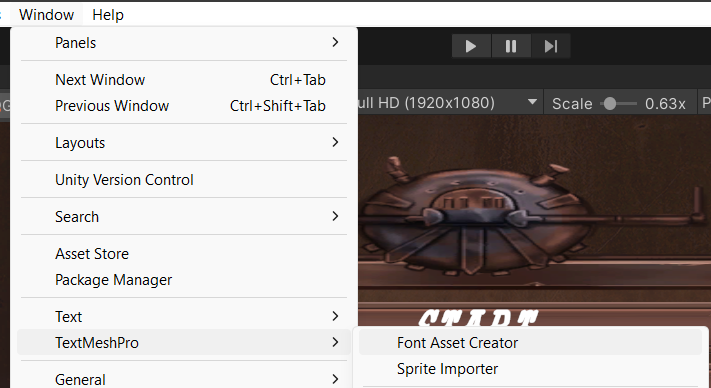
* 1. Для того щоб текст змінював свій розмір разом з кнопкою, потрібно ввімкнути властивість Auto Size. Далі задайте Min та Max розміри, яких може набувати текст. Оберіть Font Asset та Material Preset з існуючих у паці, або створіть власний шрифт та матеріал для нього та встановіть їх у властивості тексту. *Також ознайомтесь з іншими налаштуваннями тексту.*



* 1. *(Опціонально)Створення власного шрифту. Для швидкого прототипування гри можна використовувати ресурси та веб-сайти, де ви можете знайти безкоштовні шрифти.* *Перш ніж використовувати шрифти у своєму проекті, переконайтеся, що ви ознайомилися з умовами ліцензій кожного конкретного шрифту, особливо якщо це комерційний продукт. Ось кілька популярних ресурсів:*
* *Dafont;*
* *Google Fonts;*
* *Font Squirrel;*
* *1001 Free Fonts;*
* *FontSpace: FontSpace;*



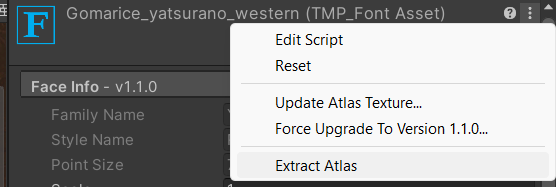
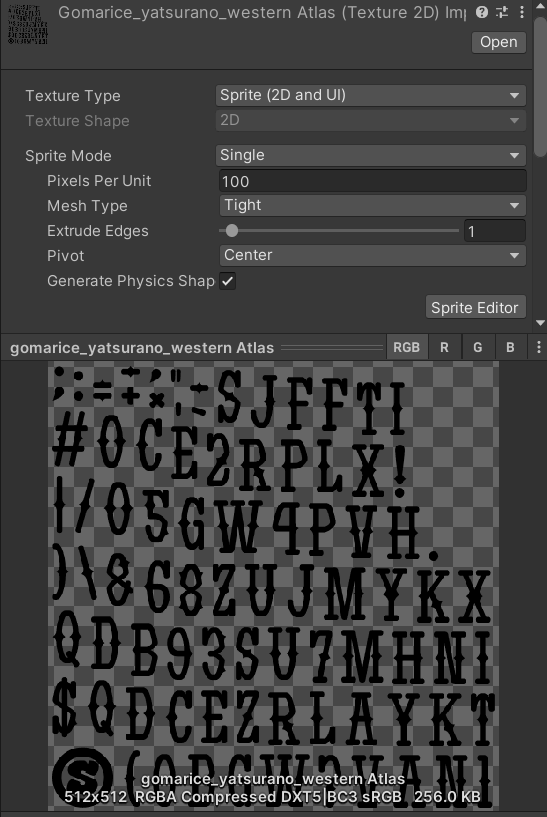
*Для демонстрації був використаний шрифт з сайту Dafont. Після того як ви скачали архів, розархівуйте його. Створіть в проекті папку Fonts, та додайти в неї шрифт. Щоб шрифт працював з TextMeshPro компонентою, перейдіть у вкладку Window > TextMeshPro > Font Asset Creator. Додайте до Source Font File скачаний шрифт. Нажміть Generate Font Atlas та збережіть його. Замініть властивість TextMeshPro Font Asset згенерованим шрифтом.*

**

*В результаті ви отримаєте бажаний шрифт, але для того щоб він виглядав цікавіше ви ще можете створити для нього Material Preset.*

**

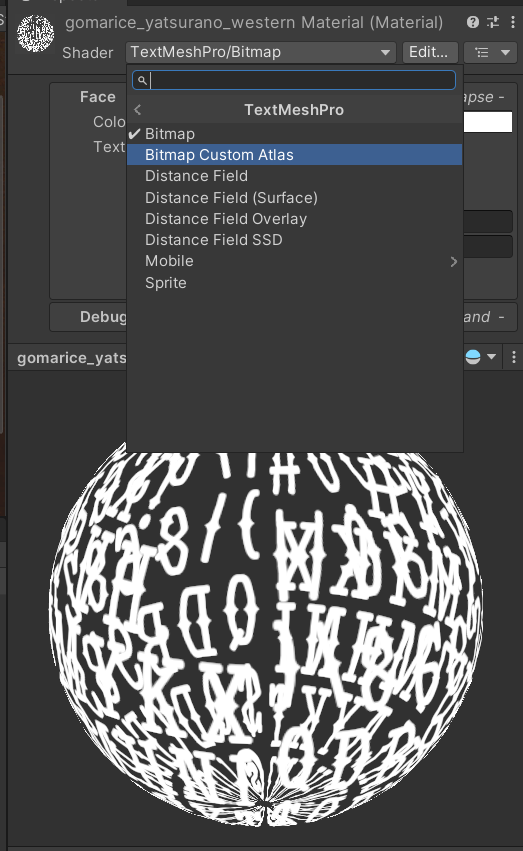
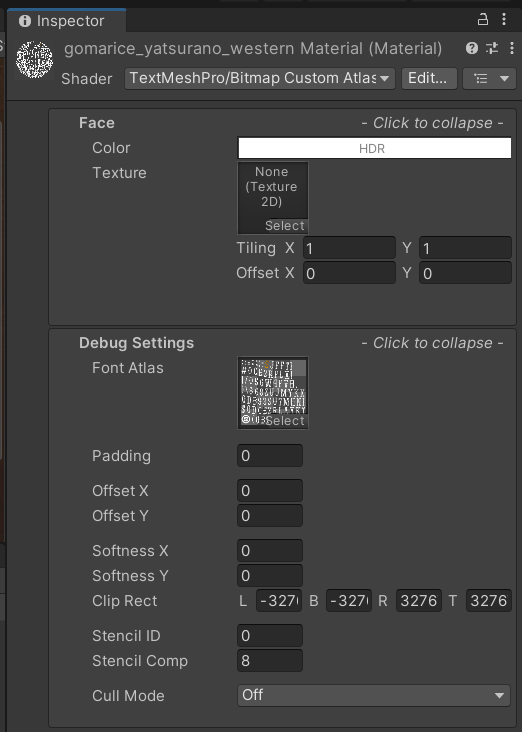
*Нажміть на згенерований шрифт. В інспекторі нажміть на три крапки та оберіть Extract Atlas. Тепер вам потрібно відкрити згенерований спрайт в будь якій програмі для роботи з графікою (PhotoShop, Aesprite, інші).*

** **

*Якщо ви збираєтесь змінювати колір шрифту в середині рушія, за допомогою комопненти, тоді краще зробити шрифт білого кольору. Далі змініть вигляд літер в графічному редакторі за бажанням.*

**

*Після того як ви попрацювали зі зображенням, імпортуйте його в юніті. Далі обираємо Font Material. В інспекторі міняємо його Shader на TextMeshPro > Bitmap Custom Atlas. Нажміть в інспекторі на Debug Settings і замініть властивість Font Atlas.*

** **

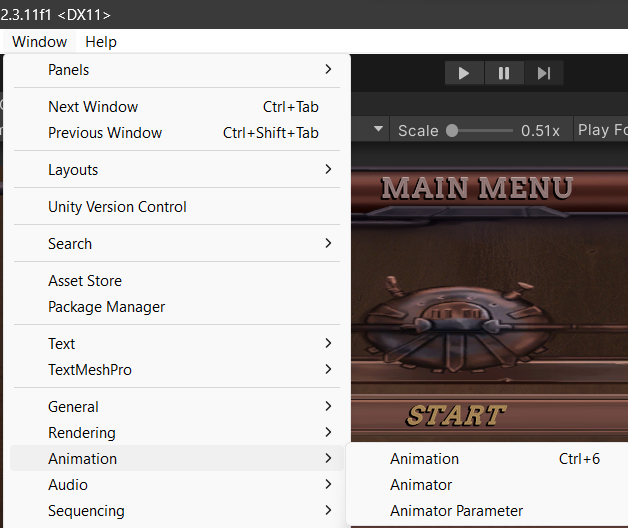
**

1. Перейдемо до створення анімацій. Відкрийте інспектор будь-якої кнопки. Компонент Button має властивість Transition, яка визначає, як буде здійснюватися перехід між різними станами кнопки (Normal, Highlighted, Pressed, Disabled). Властивість Transition визначає, як буде здійснюватися анімація при переході між цими станами.

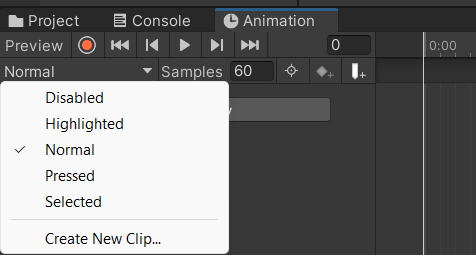
* None - немає анімації переходу. Кнопка миттєво перемикається між станами без будь-яких візуальних змін.
* Color Tint - змінюється колір кнопки при переході між станами. Це може включати зміну кольору тексту та інших елементів.
* Sprite Swap - замінюється спрайт кнопки при переході між станами. Використовується, коли вам потрібно змінювати весь вигляд кнопки.
* Animation - використовується в анімаційному кліпі для керування анімацією переходу. Можна використовувати складові анімації, такі як зміна положення, розміру, обертання тощо.

Вибір залежить від вашого досвіду, потреб проекту та особливостей анімацій, які вам потрібно виконати. Також варто пам’ятати що анімації можна створювати за допомогою скриптів. В такому випадку може бути корисне використання асету DoTween, який можна завантажити з assetsstore : <https://assetstore.unity.com/packages/tools/animation/dotween-hotween-v2-27676>

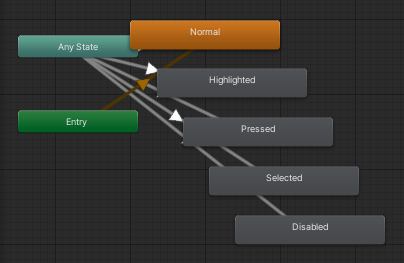
Для данної лабораторної роботи пропонується використовувати Transition Animation. Для початку роботи встановіть данний тип переходу на кнопці. Нажміть на компоненті Button кнопку Auto Generate Animation, після чого автоматично створиться Animator Controller з відповідними пустими анімаціями. При створенні Animator Controller оберіть нову папку з назвою Animations для його збереження.Для роботи над анімаціями відкрийте вікно Animation.



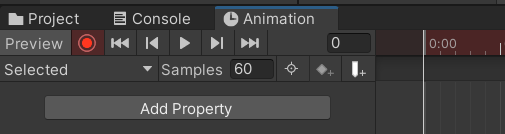
Нажміть на кнопку StartButton в ієрархії об’єктів сцени. Після цього ви побачите анімаційну шкалу у вікні Animation. Ви можете обирати яку саме анімацію хочете редагувати з випадаючого списку.



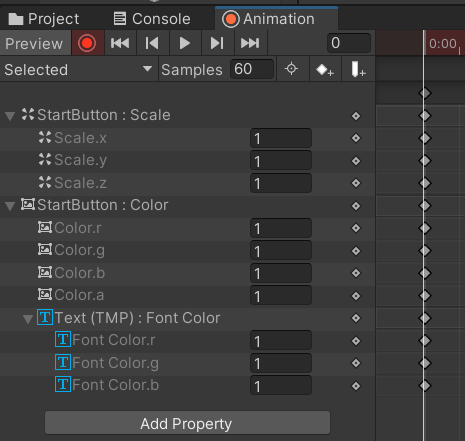
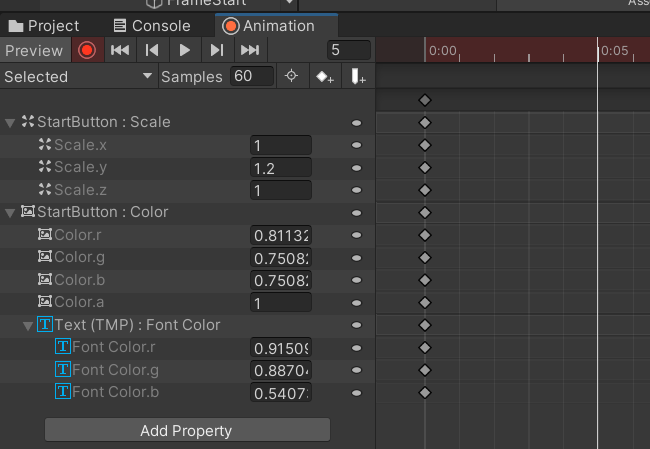
В списку видні всі анімації які автоматично були створені в контролері анімацій.

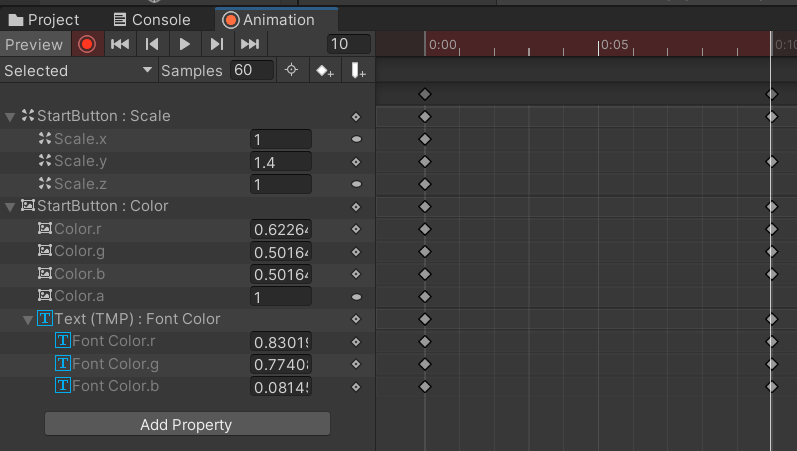


Для початку редагування анімації потрібно натиснути на червоне коло.



Поки ви не натиснули червоне коло ще раз, ви можете модифікувати параметри компонентів, які знахоться на тому ж об’єкті, що і Animator Controller комопнента. На червоній анімаційні шкалі натисніть мишкой на позначку 0:10. Тепер модифікуйте будь які значення об’єкта. Саме просто що можна зробити : змінити спрайт кнопки, збільшити її Scale, збільшити розмір тексту, змінити колір тексту і т.д. Чим більший проміжок кадрів між початковим значенням і кінцевим, тим плавнішою буду зміна значення.



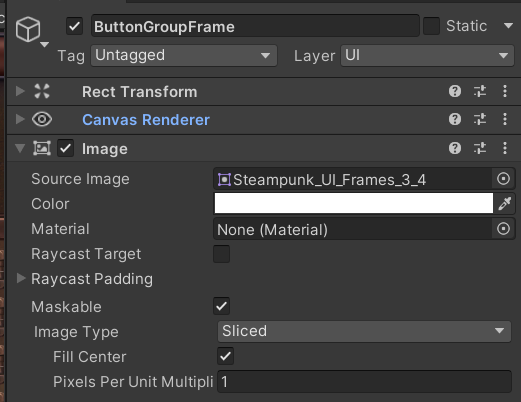
* 1. Якщо при наведенні курсора на кнопку не спрацьовує подія Selected, значить на сцені є об’єкти, які знаходяться поверх кнопки і приймають події курсору. В ієрархії об’єктів кожен наступний об’єкт знаходиться поверх попереднього.



Такий випадок можна виправити за допомогою властивості Image - Raycast Target. Raycast Target визначає, чи може графічний об'єкт (зображення або інший ігровий об'єкт) приймати події курсору.

* Якщо Image Raycast Target включено - це означає, що об'єкт реагує на кліки, наведення курсору або інші події, пов'язані з курсором.
* Якщо Image Raycast Target вимкнено - Об'єкт не приймає події курсору. Це призводить до ігнорування курсором чи лазером цього об'єкта при взаємодії.

Ця властивість може також бути корисною, коли ви хочете, наприклад, мати кнопку, яка тимчасово вимикає свою взаємодію під час анімаційного ефекту чи іншої дії, і тим самим уникнути декількох кліків під час виконання певної операції.



1. Наступним кроком буде обробка подій нажаття на кнопку. В Unity існує багато способів, щоб впоратись з цією задачею.
   1. Використання обробника подій кнопки в інтерфейсі. Спершу створіть скрипт MenuManager у папці Scripts > UI. Потім додайте до скрипта наступний код :

public class MenuManager : MonoBehaviour

{

[SerializeField] private List<GameObject> menuTabs;

public void StartGame()

{

SceneManager.LoadScene(1); // запускаємо рівень гри

}

public void OpenMenuTab(GameObject tab)

{

for(int i = 0; i < menuTabs.Count; i++)

{

if (menuTabs[i] == tab)

menuTabs[i].SetActive(true);

else

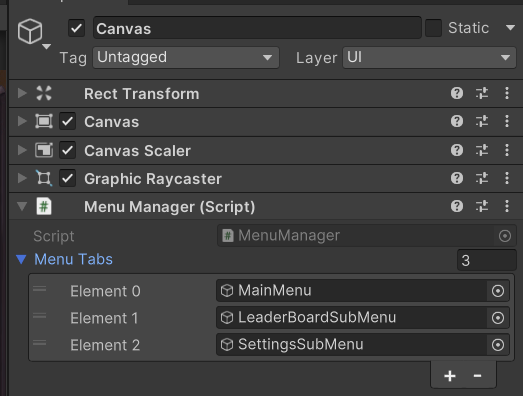
menuTabs[i].SetActive(false);

}

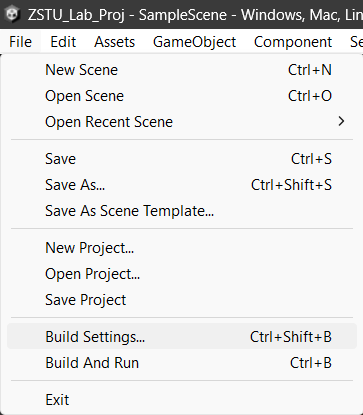
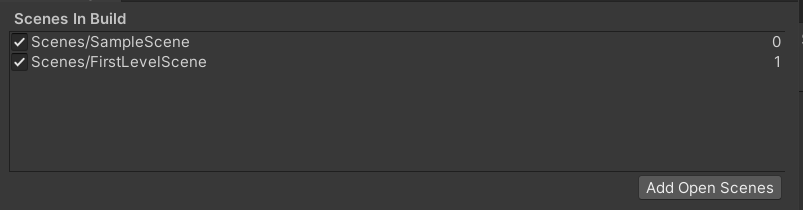
}

}

Після цього вам потрібно додати скрипт до об’єкта Canvas як компоненту. Додайте до Canvas ще декілька панелей на яких будуть побудовані підменю. Перетягніть(Drag&Drop) створені панелі до властивості menuTabs скрипта MenuManager.



Створіть ще одну сцену у папці Scene, та назвіть її FirstLevelScene. Перейдіть до створеної сцени два рази натиснувши мишкою на неї. Далі перейдіть до налаштування білда. Натисніть кнопку AddOpenScene та можете закривати вікно і переходити до сцени з меню.

Відкрийте інспектор кнопки, яка буде розпочинати гру. Знизу компоненти Button знайдіть подію OnClick та нажміть на плюс. Перетягніть Canvas до поля None(Object) та у випадаючому списку No Function оберіть MenuManager > StartGame(). Такі самі дії потрібно виконати з іншими кнопками, але для фунції OpenMenuTab() потрібно знизу передати об’єкт підменю.



Інтерфейс Unity для обробки подій кнопки є зручним і простим для багатьох випадків, але існують ситуації, коли його використання може бути менш зручним або менш ефективним. Нижче перераховані деякі можливі недоліки:

* У випадках зі складним інтерфейсом або великою кількістю кнопок може виникнути велика кількість складних моментів у керуванні логікою подій для кожної кнопки окремо.
* Робить код менш розширюваним та важче у відлагодженні.
* Обмежений визначеною подією onClick, що перешкоджає достатній гнучкості для реагування різних станів кнопки.
* Використання інтерфейсу Unity може призвести до виникнення непередбачуваних помилок
* Для складних анімацій або змін стану кнопки може знадобитися більше контролю, якого надає програмований підхід.

* 1. Для того щоб оброблювати всі події кнопки створіть скрипт ButtonHandler та додайте його на кнопку. Додайте наступний код до скрипта:

enum ButtonAction

{

StartGame,

ExitGame,

OpenMenu

}

public class ButtonHandler : MonoBehaviour, ISelectHandler, IPointerClickHandler, IPointerDownHandler, IPointerUpHandler, IPointerEnterHandler, IPointerExitHandler

{

[SerializeField] private ButtonAction buttonAction;

[SerializeField] private GameObject menuTab;

private MenuManager \_menuManager;

private void Start()

{

\_menuManager = GetComponentInParent<MenuManager>();

}

private void DoAction()

{

switch (buttonAction)

{

case ButtonAction.StartGame:

\_menuManager.StartGame(); break;

case ButtonAction.ExitGame:

\_menuManager.ExitGame(); break;

case ButtonAction.OpenMenu:

\_menuManager.OpenMenuTab(menuTab); break;

}

}

public void OnPointerClick(PointerEventData eventData)

{

Debug.Log("OnPointerClick");

DoAction();

}

public void OnPointerDown(PointerEventData eventData)

{

Debug.Log("OnPointerDown");

}

public void OnPointerEnter(PointerEventData eventData)

{

Debug.Log("OnPointerEnter");

}

public void OnPointerExit(PointerEventData eventData)

{

Debug.Log("OnPointerExit");

}

public void OnPointerUp(PointerEventData eventData)

{

Debug.Log("OnPointerUp");

}

public void OnSelect(BaseEventData eventData)

{

Debug.Log("OnSelect");

}

}

\*(Зайве?)*Цей код надає змогу керувати усіма подіями кнопки. Його можна покращити. Можна використати наслідування, зробивши цей клас базовими і реалізувавши скрипт відкриття підменю і скрипт дії. Також тут можуть бути задіяні Generic Types разом з підменю (реалізувати для них скрипти), або використаний PropertyDrawers.*

1. Додаємо до меню фонову музику та звуки взаємодії з кнопками. Для цього створіть скрипт AudioManage та додайте до нього наступний код :  
   public class AudioManager : MonoBehaviour

{

[SerializeField] private AudioClip themeClip;

[SerializeField] private AudioClip clickClip;

[SerializeField] private AudioClip highlitedClip;

private List<AudioSource> \_soundSources;

private GameObject \_sourcesHolder;

// clip volume , looped додати

public void PlaySound(SoundType soundType)

{

bool playedFlag = false;

foreach (AudioSource source in \_soundSources)

{

if (!source.isPlaying)

{

playedFlag = true;

source.clip = ChooseClip(soundType);

source.Play();

break;

}

}

if (!playedFlag)

{

CreateNewSource();

PlaySound(soundType);

}

}

private void Start()

{

\_soundSources = new List<AudioSource>();

\_sourcesHolder = new GameObject("AudioSourcesHolder");

\_sourcesHolder.transform.parent = this.transform;

PlaySound(SoundType.theme);

}

private AudioClip ChooseClip(SoundType soundType)

{

switch (soundType)

{

case SoundType.click:

return clickClip;

case SoundType.highlited:

return highlitedClip;

case SoundType.theme:

return themeClip;

}

return null;

}

private void CreateNewSource()

{

GameObject newSource = new GameObject("AudioSource" + \_soundSources.Count);

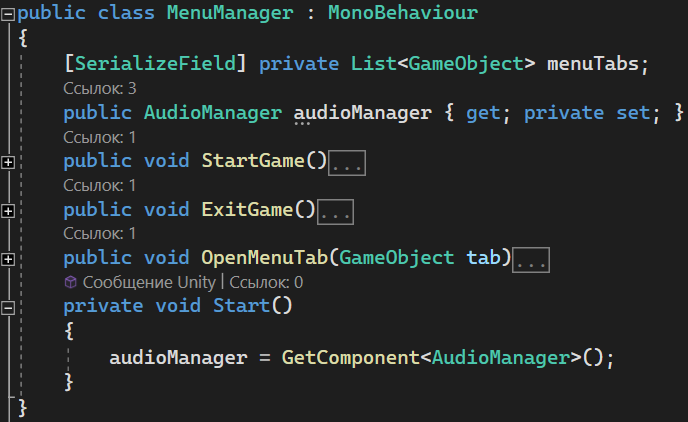
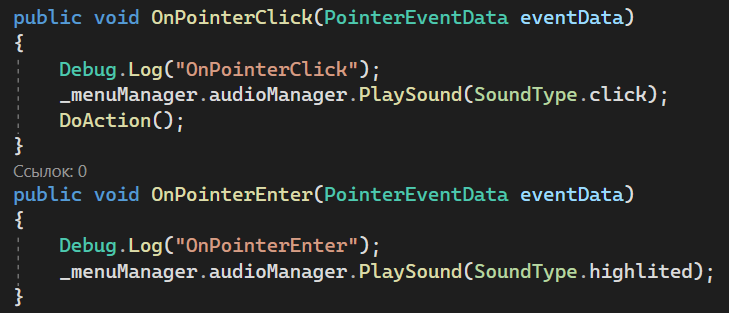
newSource.transform.parent = \_sourcesHolder.transform;

\_soundSources.Add(newSource.AddComponent<AudioSource>());

}

}

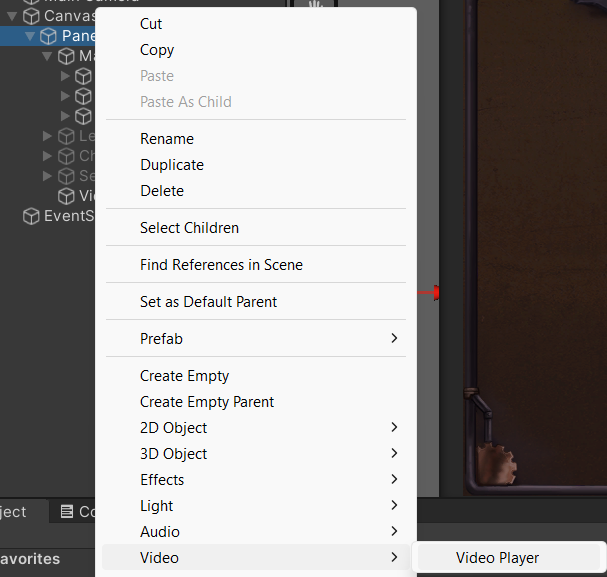
Додайте щойно створений скрипт на об’єкт Canvas. Скачайте ассети звуків для меню з assetsstore : <https://assetstore.unity.com/packages/audio/sound-fx/ui-sound-effects-collection-pack-2-buttons-27803> . Заповніть пусті поля компоненти AudioManager якимось звуками. *Для поля themeClip краще знайти трек який можна зациклити. Цей трек буде фоновою музикою.* Модифікуйте скрипти MenuManager та ButtonHandler.

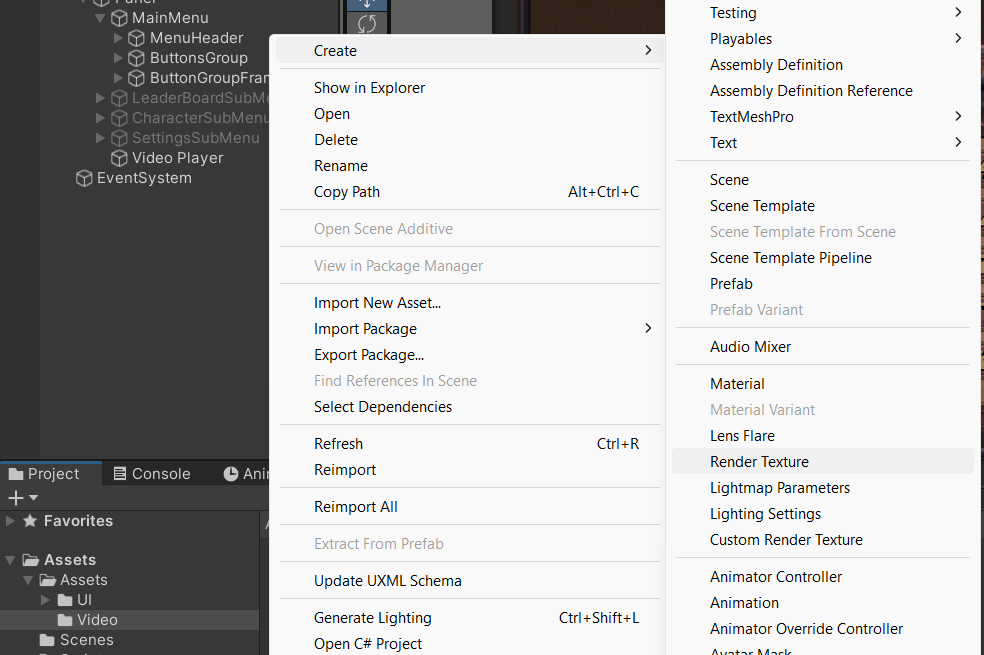
1. Для того щоб зробити анімований фон можна розмістити елементи, які мають компонент Sprite Renderer, та занімувати їх схожим способом як і кнопки, але це буде дуже об’ємний процес який займатиме багато часу. Швидший спосіб це використання відео кліпу, який можна зациклити. Скачайте підходящий для вашого меню відео кліп на одному із сайтів:

* Pexels Videos
* Videvo
* Pixabay
* Coverr

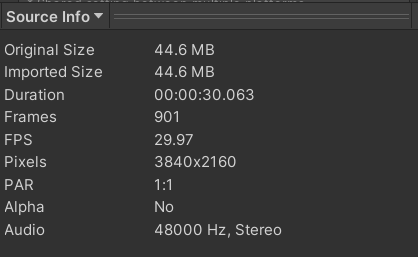
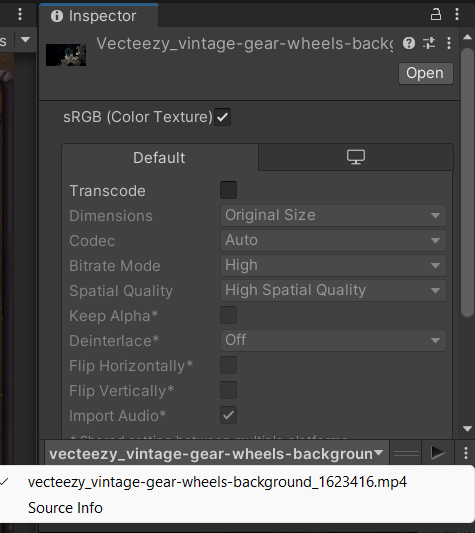
Додайте відеокліп до проекту. Додайте до панелі новий об’єкт, який містить компонент VideoPlayer. Замініть на панелі компонент Image на RawImage.



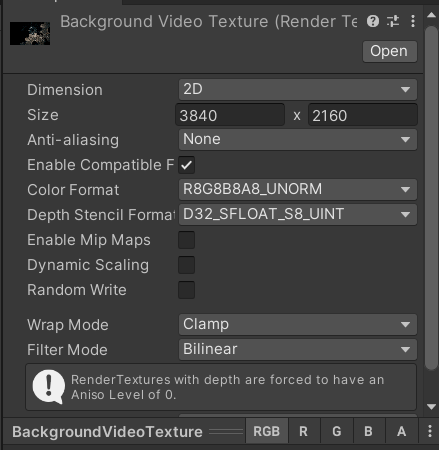
В папці з відеокліпом створіть RenderTexture.



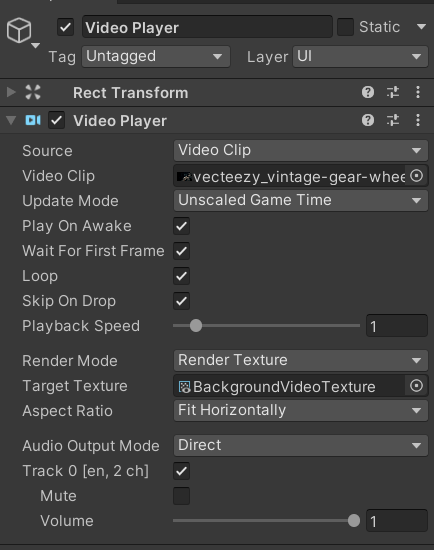
Відкрийте інспектор завантаженого відеокліпа. Нажміть на назву відео кліпа та оберіть SourceInfo, щоб подивитись інформацію про розширення відеокліпа. В стрічці Pixels ви його зможете побачити.



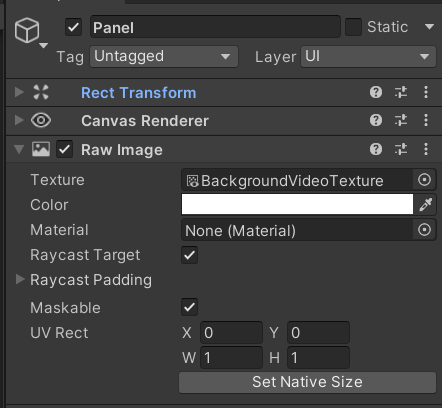
Задайте відповідне розширення у створеній RenderTexture



Налаштуйте компоненту VideoPLayer

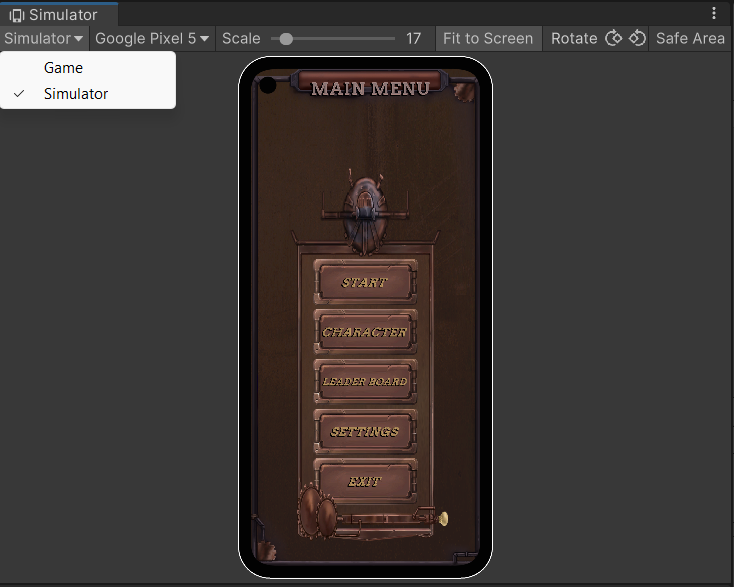


Надайте полю Texture, компоненти RawImage, налаштовану RenderTexture



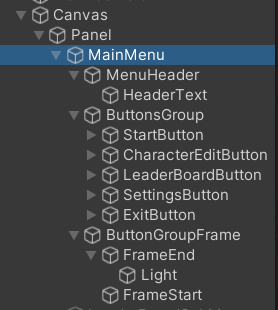
Таким чином можна також реалізувати Intro для вашої гри.

1. Перевірте виконання завдань за допомогою Simulator. Оберіть декілька пристроїв з різним розширенням і протестуйте програму. Ви також можете завантажити додаткові розширення.

Завдання :

1. Додайте ще кнопок до меню та створіть підменю. Завершіть стилізувати меню за своїм смаком.

1. Додайте до підменю такі елементи Slider,ScrollView.
2. Зробіть сцену авторизації. Додайте до сцени необхідний UI.
3. Удоскональте скрипти MenuManager,ButtonHandler,AudioManager.
4. Додайте та налаштуйте AudioMixerGroup.