**Unity**

Unity — кроссплатформенный ігровий движок з вбудованою IDE, розроблений Unity Technologies інструмент для розробки дво- та тривимірних додатків та ігор.

IDE означає інтегроване середовище розробки: Unity насправді є об'єднанням

(1) ігровий движок, який дозволяє створення запуску гри в різних середовищах

(2) додаток, в якому «видимі частини» гри можуть бути об'єднані (IDE) з графічним попереднім переглядом і з використанням контрольованої функції «плей»

(3) a **code editor**

Створені за допомогою Unity приложения працюють під системами Windows, OS X, Android, Apple iOS, Linux[3], а також на ігрових консолях Wii, PlayStation 3 і XBox 360.

Є можливість створювати інтернет-додатків за допомогою спеціального під'єднуваного модуля для браузера Unity, а також за допомогою експериментальної реалізації в межах модуля Adobe Flash Player[4]. Додатки, створені за допомогою Unity, підтримують DirectX та OpenGL.

**Технічні характеристики**

Ігрова логіка пишеться за допомогою C#, раніше так була можливость використовувати Boo та JavaScript, але розробники відмовились від їх підримки.

Ігровий рушій повністю пов'язаний із середовищем розробки. Це дозволяє випробовувати гру прямо в редакторі;

Робота з ресурсами можлива через звичайний Drag&Drop.

Підтримка імпортування великої кількості форматів файлів;

Вбудований генератор ландшафтів;

Вбудована підтримка мережі;

Інтерфейс редактора дуже гнучкий, є можливість писати свої вікна редактора та всіляки розширення для нього.

Існує рішення для спільної розробки — Asset Server.

Тому, коли у вас є ігровий дизайн і все необхідне, ви можете зібрати графіку, звуки, анімацію в IDE Unity, написати в редакторі код, пов'язаний з ресурсами, а потім згенерувати іграбельность додатоку, який працює в декількох різних середовищах, як одна з особливостей Unity є те, що він мультиплатформенний, тобто більш-менш та ж сама гра може працювати аналогічно на iPad або на Windows, скажімо.

Процес, який я тільки що описав, по суті своїй досить складний - відеоігри є складні структури. Unity надає повний робочий процес і багато допомоги, а також великий набір функцій.

**Unity 2D**

Багато 2D ігор використовують плоску графіку, так звані спрайти, які немають тривимірної геометрії. Спрайт відображаються на екрані як плоскі картинки, а камера при цьому не має перспективи. Для таких ігор вам слід вибрати режим 2D.

**2D Графіка**

2D об'єкти називаються спрайт. По суті, спрайт є стандартною текстурою, але для більш ефективного об'єднання і управління ними в процесі розробки, є спеціальні техніки. Unity надає вбудований Редактор спрайт, що дозволяє витягти спрайт з великого зображення. Він дозволяє редагувати компоненти зображень всередині єдиної текстури. Вам слід його використовувати, наприклад, для поділу рук, ніг і тіла персонажа всередині одного зображення.

**2D Фізика**

Unity має окремий движок для 2D фізики, що робить оптимізацію доступною тільки для 2D об'єктів. Його компоненти відповідають стандартним компонентам 3D фізики, таким як Rigidbody, Box Collider і Hinge Joint, але з додаванням до імені "2D". Так, спрайт можуть бути додані компоненти Rigidbody 2D, Box Collider 2D і Hinge Joint 2D. Більшість 2D компонентів є просто "сплющені" версії відповідних 3D компонентів (наприклад, Box Collider 2D - квадрат, а Box Collider - куб), однак і тут є кілька винятків.

**2D геймплей з тривимірною графікою**

Деякі ігри використовують тривимірні моделі для персонажів і оточення, але при цьому обмежують геймплей двома вимірами. Наприклад: камера може використовувати "бічне ковзання" ( "side scrolling view") і гравець може рухатися тільки в двох вимірах, але гра як і раніше використовує тривимірні моделі в якості перешкод, а камера має перспективу. Для таких ігор 3D виконує швидше стилістичну роль, ніж функціональну. Такий тип ігор іноді називають "2.5D". Хоча геймплей і двомірний, ви все одно будете керувати тривимірними моделями, тому для створення такої гри слід вибирати режим 3D.

**2D геймплей і графіка з використанням перспективної камери**

Це ще один популярний стиль двомірних ігор, що використовує 2D графіку, але з перспективної камерою, щоб створити ефект паралакса (зміщення) при русі. Це сцена в стилі "Картонний театр", де вся графіка є двомірної, але вона розташована на різній відстані від камери. Найбільш ймовірно, що для такої гри вам підійде режим 2D, проте у вашій грі потрібно буде переключити режим камери на перспективну (Perspective) і відображення сцени в 3D режим.

**Фізика**

Щоб фізичне поведінка була правдоподібною, об'єкт в грі потрібно правильно прискорити і задіяти зіткнення, гравітацію і інші сили. Вбудований в Unity фізичні движки забезпечують вас компонентами для обробки симуляції фізики. За допомогою налаштування всього декількох параметрів, можна створити об'єкти, які ведуть себе пасивно реалістично (тобто, вони будуть переміщені в результаті зіткнень і падінь, але не почнуть рухатися самі по собі). Керуючи фізикою з скриптів, ви можете надати об'єкту динаміку автомобіля, машини або навіть рухомого шматка тканини.

Розрахунки фізики виконує фізичний движок PhysX від NVIDIA. PhysX - сполучне програмне забезпечення (англ. Middleware), багатоплатформовий фізичний движок для симуляції ряду фізичних явищ, а також комплект засобів розробки (SDK) на його основі.

Підпрограмне забезпечення PhysX SDK дозволяє розробникам ігор уникати написання власного програмного коду для обробки складних фізичних взаємодій в сучасних комп'ютерних іграх. PhysX SDK може використовуватися не тільки в середовищі Microsoft Windows, але і також в Linux, проте підтримка процесора PhysX поки працює тільки для Windows.

На відміну від більшості інших фізичних движків, які поставляються і встановлюються разом з грою, PhysX SDK необхідно встановити окремо. Він встановлюється як окремий драйвер. Якщо на комп'ютері встановлена ​​плата PhysX, то драйвер PhysX SDK при роботі буде використовувати її ресурси. Якщо ж PhysX відсутня, то обчислювальні завдання будуть переноситися на центральний процесор.

Фізичний движок PhysX SDK складається з трьох головних компонентів з обробки фізики:

обробка твердих тіл (англ. rigid body);

обробка тканин (англ. cloth);

обробка рідин (англ. fluid);

Примітка: Насправді в Unity є два окремих фізичних движка, один для 3D фізики і один для 2D фізики. Основні поняття ідентичні в обох двигунах (за винятком додаткового виміру в 3D), але вони реалізовані з різними компонентами. Так, наприклад, існує компонент Rigidbody для 3D фізики і аналогічний Rigidbody 2D для 2D фізики.

**Rigidbodies (тверді тіла)**

Rigidbody - це основний компонент, що включає фізичне поводження для об'єкта. З прикріпленим Rigidbody, об'єкт негайно почне реагувати на гравітацію. Якщо доданий один або кілька компонентів Collider, то при колізіях (зіткненнях) об'єкт буде пересуватися.

Так як компонент Rigidbody управляє переміщенням об'єкта, до якого він прикріплений, вам не слід намагатися впливати на об'єкт з коду за допомогою зміни таких властивостей Transform, як position і rotation. Замість цього вам слід застосовувати сили для того, щоб штовхати об'єкт і дозволити фізичному движку розрахувати результати.

Існують випадки, при яких ви можете захотіти, щоб у об'єкти був Rigidbody, але фізичний движок не керував його рухом. Наприклад, ви можете побажати управляти своїм персонажем безпосередньо з коду, але при цьому хочете, щоб він взаємодіяв з тригерами (див. Тригери нижче). Цей тип не фізичного руху, вироблений за допомогою коду, називається кинематическим рухом. У компонента Rigidbody є властивість Is Kinematic, яке може виключити об'єкт з-під контролю фізичного движка, і дозволити переміщати його кінематично з скрипта. Значення Is Kinematic можна міняти з коду, щоб ввімкнути або вимкнути фізику для об'єкта, але ця можливість вимагає додаткових ресурсів і повинна використовуватися як можна рідше.

**Колайдери (Colliders)**

Компоненти коллайдера визначають форму об'єкта для цілей фізичних зіткнень. Коллайдер, який невидимий, не обов'язково повинен мати ту ж форму, що і сітка об'єкта, і насправді грубе наближення часто більш ефективно і невиразно в ігровому процесі.

Найпростішими (і найменш ресурсоємними) колайдер є так звані примітивні типи коллайдеров. У 3D це Box Collider, Sphere Collider і Capsule Collider. У 2D ви можете використовувати Box Collider 2D і Circle Collider 2D. Будь-яке їх кількість може бути додано до одного об'єкту для створення складових коллайдеров.

При акуратному позиціонуванні і визначенні розмірів складові колайдери часто можуть досить добре наближатися до форми об'єкта, зберігаючи при цьому низькі витрати процесора. Додаткову гнучкість можна отримати, якщо мати додаткові колайдери на дочірніх об'єктах (наприклад, ящики можна повертати щодо локальних осей батьківського об'єкта). При створенні складеного коллайдера, подібного до цього, повинен бути тільки один компонент Rigidbody, поміщений в кореневій об'єкт в ієрархії.

Колайдери можна додавати до об'єкта без компонента Rigidbody для створення підлог, стін і інших нерухомих елементів сцени. Вони називаються статичними коллайдера. Загалом, ви не повинні міняти положення статичних коллайдеров, змінюючи положення Transform, так як це сильно вплине на продуктивність фізичного движка. Колайдери на об'єкті, який дійсно має Rigidbody відомі як динамічні колайдери. Статичні колайдери можуть взаємодіяти з динамічними коллайдера, але оскільки у них немає твердого тіла, вони не будуть рухатися у відповідь на зіткнення.

**З’єднання (****Joints)**

За допомогою компонента Joint, ви можете додати один rigidbody об'єкт до іншого або до фіксованої точки в просторі. Зазвичай ви бажаєте, щоб у зчленування була хоч якась свобода руху, так що Unity надає різні Joint компоненти, що реалізують різні обмеження.

Наприклад, шарнірне з'єднання дозволяє обертати навколо певної точки і осі, в той час як пружинне з'єднання утримує об'єкти один від одного, але дозволяє злегка розтягнути відстань між ними.

Компоненти 2D сполуки мають 2D в кінці назви, наприклад, шарнірне з'єднання 2D. Див. З'єднання 2D для зведення 2D з'єднань і корисної довідкової інформації.

У зчленувань також є інші опції, які можна включити для різних ефектів. Наприклад, ви можете налаштувати зчленування так, щоб воно розривалося при застосуванні до нього сили більше заданого порогу. Деякі зчленування також дозволяють виникати рушійну силу (drive force) між сполученими об'єктами для автоматичного приведення їх у рух.

**Контролери персонажа (Character Controllers)**

Персонажу в грі від першого або третього лиця часто потрібна деяка фізика заснована на зіткненнях, щоб він не падав крізь підлогу або не ходив крізь стіни. Зазвичай, все ж прискорення і рух персонажа фізично нереалістичні, таким чином він може прискорюватися, гальмувати і змінювати напрямок миттєво, не наражаючись впливу імпульсу.

У 3D фізики цей тип поведінки можна створити використовуючи Character Controller. Цей компонент дає персонажу простий коллайдер в формі капсули, який завжди знаходиться у вертикальному положенні. У Controller'а є свої особливі функції для призначення швидкості і напряму об'єкта, але, на відміну від справжніх коллайдеров, він не вимагає Rigidbody і його ефекти імпульсу нереалістичні.

Character Controller не може проходити крізь Static колайдери в сцені, тому він буде рухатися по підлозі і блокуватися стінами. Під час руху він може відштовхувати Rigidbody об'єкти, але входять зіткнення не впливатимуть на його прискорення. Це означає, що ви можете використовувати стандартні 3D колайдери, щоб створити сцену по якій буде ходити Controller, але ви не обмежені реалістичним, з точки зору фізики, поведінкою самого персонажа.

# Scripting

Сценарії, які використовуються в усіх компонентах, створюють у Unity. Більшість запропонованих потрібних скриптів, щоб реагувати на вихідні дані від ігрока та організувати зміни в ігровому процесі, коли це необхідно. Крім того, сценарії можуть використовувати для створення графічних ефектів, управління фізичною поведінкою об'єктів або навіть реалізації власної системи

Поведінка GameObjects контролюється Компонентами які прив'язані до них. Хоча вбудовані компоненти Unity можуть бути дуже універсальними, ви скоро побачите, що вам потрібно вийти за рамки того, що вони можуть надати для реалізації ваших власних ігрових функцій. Unity дозволяє вам створювати свої власні Компоненти, використовуючи сценарії. Це дозволяє вам запускати ігрові події, змінювати властивості компонента з плином часу і реагувати на введення користувача будь-яким зручним для вас способом.

**Аудіо Огляд**

Гра була б неповною без будь-якого звуку, будь то фонова музика або звукові ефекти. Аудіосистема Unity є гнучкою і потужною. Він може імпортувати більшість стандартних форматів аудіофайлів і має складні функції для відтворення звуків в тривимірному просторі, при необхідності з такими ефектами, як відлуння і фільтрація. Unity також може записувати звук з будь-якого доступного мікрофона на комп'ютері користувача для використання під час гри або для зберігання і передачі.

Unity може імпортувати звукові файли форматів AIFF, WAV, MP3 і Ogg так само, як і інші ресурси, просто перетягуючи файли на панель «Проект». При імпорті аудіофайлу створюється Audio Clip який потім можна перетягнути на джерело звуку або використовувати з сценарію.

**Відео огляд**

Використовуйте відеосистему Unity для інтеграції відео в вашу гру. Відеозапис може додати реалізму, заощадити при рендеринге складність або допоможе вам інтегрувати контент, доступний ззовні.

**АНІМАЦІЯ**

Можливості анімації Unity включають в себе анімацію з можливістю перенастроювання, повний контроль ваги анімації під час виконання, виклик подій зсередини відтворення анімації, складний кінцевий автомат

  ієрархії і переходи, змішані фігури для лицьової анімації і багато іншого.

**Огляд системи анімації**

Unity має багату і складну систему анімації (іноді її називають «Mecanim»). Це забезпечує:

Простий робочий процес і настройка анімації для всіх елементів Unity, включаючи об'єкти, персонажів і властивості.

Підтримка імпортованих анімаційних кліпів і анімації, створеної в Unity

гуманоїдна анімація

 ретаргетінг

 - можливість застосовувати анімацію від однієї моделі персонажа до іншого.

Спрощений робочий процес для вирівнювання анімаційних кліпів.

Зручний попередній перегляд анімаційних кліпів, переходів і взаємодій між ними. Це дозволяє аніматорам працювати більш незалежно від програмістів, створювати прототипи і переглядати свої анімації до того, як код ігрового процесу буде підключений.

Управління складними взаємодіями між анімаціями за допомогою інструменту візуального програмування.

Анімація різних частин тіла з різною логікою.

Особливості нашарування і маскування

**анімаційні кліпи**

Анімаційні кліпи є одним з основних елементів системи анімації Unity. Unity підтримує імпорт анімації з зовнішніх джерел і пропонує можливість створювати анімаційні кліпи з нуля в редакторі за допомогою вікна «Анімація».

**контролери аніматорів**

Аніматор Контролер дозволяє вам організувати і підтримувати набір анімацій для персонажа або іншого анімованого ігрового об'єкта.

Контролер має посилання на анімаційні кліпи

 використовуваному в ньому, і управляє різними станами анімації і переходами між ними за допомогою так званого кінцевого автомата, який можна розглядати як свого роду блок-схему або просту програму, написану на мові візуального програмування в Unity.

**XR - це загальний термін, який включає в себе наступні типи додатків:**

Віртуальна реальність

  (VR): додаток імітує абсолютно інше середовище навколо користувача.

Змішана реальність (MR): додаток об'єднує своє власне середовище з реальним середовищем користувача і дозволяє їм взаємодіяти один з одним.

Доповнена реальність (AR): додаток розміщує контент поверх цифрового уявлення про реальний світ.

**Офіційно підтримувані платформи**

Unity тісно співпрацює зі стратегічними партнерами, щоб переконатися, що творці мають всю необхідну підтримку при розробці контенту для підтримуваних платформ Unity. Це включає в себе глибоку інтеграцію платформи, поліпшення движка Unity і оптимізацію стека XR для кожної платформи.

Починаючи з 2019.3, Unity офіційно підтримує такі платформи для XR:

ARKit

ARCore

Microsoft HoloLens

Windows Mixed Reality

Magic Leap

Oculus

PlayStation VR

Valve використовує XR SDK Unity для розробки свого плагіна OpenVR

**Каркас плагіна Unity XR**

Unity розробила новий плагін фреймворк під назвою XR SDK, який дозволяє провайдерам XR інтегруватися з движком Unity і в повній мірі використовувати його функції. Цей підхід на основі плагінів покращує здатність Unity швидко виправляти помилки, поширювати поновлення SDK від партнерів по платформі і підтримувати нові пристрої і середовища виконання XR без необхідності змінювати ядро.

**Вбудовані інтеграції:**

Windows Mixed Reality (HoloLens & Windows MR devices)

Oculus Android & Oculus Desktop

OpenVR (Desktop)

Google VR Android & Google VR iOS

Vuforia Engine AR (1**)**

**Vuforia Engine в Unity**

Починаючи з 2017 року, Unity інтегрує Vuforia Engine, що ще більше спрощує створення передових можливостей доповненої реальності.

**Про Vuforia**

Vuforia - це платформа доповненої реальності і інструментарій розробника програмного забезпечення доповненої реальності (Software Development Kit - SDK) для мобільних пристроїв, розроблені компанії Qualcomm. Розробники можуть легко додати розширену функціональність комп'ютерного зору в будь-який додаток, дозволяючи йому розпізнавати зображення і об'єкти і взаємодіяти з просторами в реальному світі.

Платформа Vuforia Engine підтримує розробку додатків AR для пристроїв Android, iOS і UWP.