Worktest Unity

Purple Tree - Programación

Introducción

El objetivo de este worktest es evaluar la resolución de una mecánica de gameplay con una integración de HUD básica y la exposición de los valores necesarios para ser configurados por un level designer.

El ejercicio consta de unos requerimientos mínimos y una serie de agregados opcionales, funcionales y no funcionales.

Para la resolución se debe usar Unity 2019.4.26 o Unity 2020.3.27, y puede ser entregada en un .zip o subida a un repositorio en Github. Tiene que poder probarse desde el editor y adicionalmente incluir un ejecutable para Windows o Mac.

Los assets gráficos a usar se pueden descargar de https://drive.google.com/file/d/186rsAsS36AQu4PZ4dlG-1VPDTfRQ9yrA. Se pueden usar assets adicionales de considerarlo necesario.

Definición del ejercicio

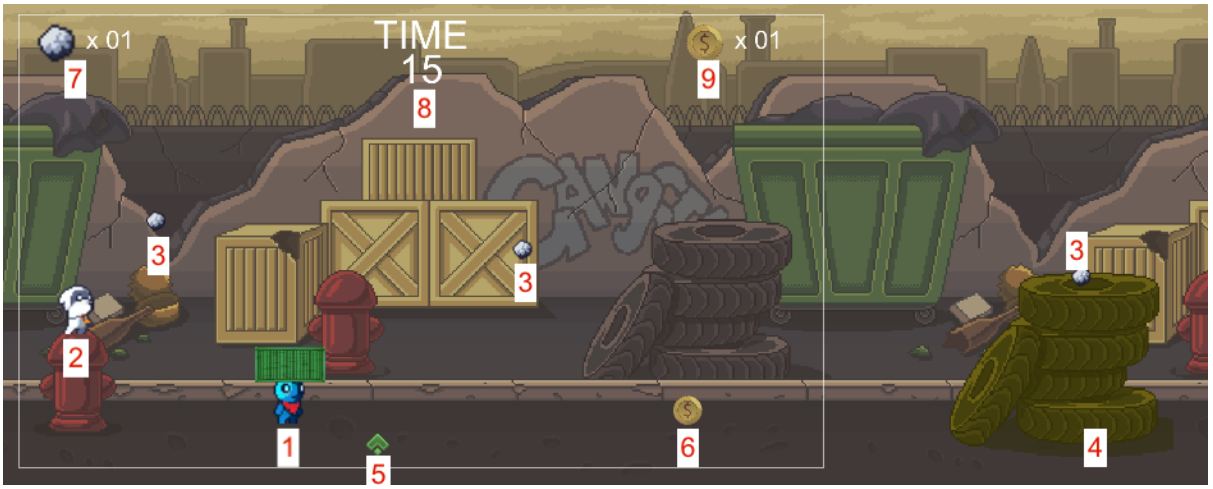
Descripción general

Se requiere implementar la siguiente mecánica en 2D:

El jugador controla a un personaje (**héroe**) que tiene una "**caja-resorte**" en la cabeza y tiene que **embocar** la mayor cantidad de **rocas** posible en el hueco superior de unas cubiertas apiladas (**meta**) al tiempo que recolecta la mayor cantidad de monedas posible antes de que se acabe el tiempo.

El **héroe** puede moverse hacia los costados dentro del **área de juego**.
Hay un segundo personaje (**lanzador**) que está fijo en un extremo de la pantalla. El **lanzador** dispara cada cierto tiempo una **roca**, que hace una parábola y cae al suelo o rebota en la **caja-resorte** del **héroe**. Dependiendo de la fuerza del lanzamiento se puede necesitar de más de un rebote para que la **roca** llegue a la **meta**.
Al **embocar** una cantidad determinada de **rocas** aparece en el **área de juego** una **moneda** que el **héroe** puede recoger al tocarla. Si pasado un tiempo la **moneda** no es recogida desaparece.

[Opcional] Una **guía** se muestra en el suelo, indicando dónde debe ubicarse el héroe para que la roca más próxima al suelo rebote justo en el centro de la **caja-resorte**.



Mockup
(imagen original: https://drive.google.com/file/d/1ZLlG5qG0P5Ygrv95RGt0r1fR-l3PJwlx)

Referencias del mockup:

1. Héroe
2. Lanzador
3. Rocas
4. Meta (el área exacta donde debe caer la roca es el hueco superior, la imagen muestra una roca de referencia en esa ubicación)
5. Guía de ubicación del héroe para rebotar la próxima roca
6. Moneda
7. HUD: Cantidad de rocas embocadas
8. HUD: Tiempo restante
9. HUD: Monedas recolectadas

Descripción detallada

Física

La física para controlar el movimiento de las rocas tiene que ser programada manualmente, no se puede usar la física provista por Unity mediante Rigidbody.

La caja-resorte tiene un determinado **factor de rebote vertical**, que indica qué porcentaje de la velocidad vertical se mantiene al rebotar. Si es menor a 1, va a perder altura con cada rebote. Si es 1, va a llegar a la misma altura después de cada rebote. Si es mayor a 1, va a ganar altura con cada rebote.

(opcional) La caja puede tener también un **factor de rebote horizontal** , que funciona igual al vertical pero afecta sólo a la velocidad en el eje horizontal. Si bien físicamente no tiene sentido que la velocidad horizontal aumente, a los fines de este ejercicio va a ser posible.

Los siguientes parámetros tienen que ser configurables por un level designer:

- Gravedad
- Factor de rebote vertical
- (opcional) Factor de rebote horizontal

Héroe

Controles:

- Teclado: Flechas izquierda-derecha

Los siguientes parámetros tienen que ser configurables por un level designer:

- Velocidad
- (opcional) Inercia al arrancar y frenar

Lanzador

Cada lanzamiento tiene un ángulo y una velocidad (speed) aleatorios dentro de rangos definidos.

Los siguientes parámetros tienen que ser configurables por un level designer:

- Ángulo de lanzamiento [min-máx]
- Velocidad (speed) de lanzamiento [min-max]
- Tiempo entre lanzamientos (min-max)

Rocas

- Las que caen al suelo desaparecen después de un tiempo

Embocar

- La roca debe caer siempre en el centro del hueco superior de la meta
- El rebote que resultaría en que la piedra se pase de la meta antes de volver a tocar el suelo se considera el **último rebote**
- Para este último rebote se ignora la configuración que se tenga de factores de rebote y, en cambio, se asigna una velocidad tal que el movimiento la lleve a caer exactamente en el hueco de la meta

Cámara

- El área de juego es más grande que el viewport de la cámara.
- La cámara debe scrollear siguendo al jugador pero no mostrar más allá de lo presentado en el mockup.

Monedas

- Aparecen en una posición al azar en el área del juego, evitando hacerlo sobre el héroe.

Los siguientes parámetros tienen que ser configurables por un level designer:

- Cantidad de rocas a embocar para que aparezca
- Tiempo antes de desaparecer

Animaciones

Héroe

- "idle": cuando está quieto
- "walk": cuando se mueve

La caja-resorte no tiene que acompañar el movimiento vertical de la animación del personaje. Este objeto debería quedar siempre en la misma posición en el eje vertical aunque se "vea raro" junto con el movimiento del héroe y parezca que está flotando en vez de estar apoyado sobre su cabeza.
(Opcional) El héroe deja un pequeño rastro de humo al comenzar a moverse.

Lanzador

- "idle": cuando espera entre lanzamientos
- "throw": cuando lanza una roca

(Opcional) La roca debería aparecer en el momento en que la animación llega al frame que se muestra en el mockup y en la posición de la mano del lanzador.

(Opcional) Guía de ubicación de héroe

Se muestra una guía en el suelo que indica dónde debe ubicarse el héroe para que la roca más próxima a caer rebote justo en el centro de la caja-resorte.

HUD

- Tiempo: En reversa, al llegar a 0 termina el nivel
- Cantidad de rocas: aumenta al embocar cada roca
- Monedas: aumenta al recoger cada moneda

    - (Opcional) Al recoger una moneda no aumentar el valor en el HUD inmediatamente. En cambio, mostrar una animación de una moneda volando desde la posición en la que fue recogida hasta la posición del icono en el HUD y, al llegar, actualizar el contador.

Los siguientes parámetros tienen que ser configurables por un level designer:

- Duración del nivel
- (Opcional) Parámetros para la animación de la moneda recogida

(Opcional) Niveles de dificultad

Se puede elegir entre distintos niveles de dificultad (fácil, difícil), cada uno con sus valores particulares para los distintos parámetros configurables.

(Opcional y posiblemente subjetivo) Configuración user-friendly

Presentar la configuración del nivel de manera intuitiva y clara para el level designer.