UNIVERSIDAD CATÓLICA DE TEMUCO
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
FACULTAD DE INGENIERÍA

Haru Monogatari

Elaborando un Videojuego en
Python y Pygame

Autores: Soto Contreras Jorge Eduardo
Verdugo Troncoso Christian Yostin Alexander

Asignatura: Programación 1, Sección 1 y 2, 2022.

Docente: Lincolao Venegas Ignacio Nicolás

Fecha: Temuco, 22 de Junio de 2022

ÍNDICE

RESUMEN

El crear un videojuego en pygame tiene sus peculiaridades , el hecho de usar herramientas que detectan rectángulos, detectar sus colisiones , crear un mapa de tiles con una aplicación para ello y traspasarlo a código , unir partes distintas, agregar menús todo esto es una experiencia que si bien es un tanto pesada para 2 personas , el lograr llevar a cabo este proyecto nos deja un aprendizaje bastante duro de como hacer un videojuego pero no de una mala manera. ya que se necesita más de 2 personas para ellos , poder dividir el trabajo de forma equitativa , tener a alguien que pueda crear imágenes de los personajes y props que ayuden a complementar el ambiente , y diseñar tanto los fondos como los menús. Este informe es la recopilación de una corta pero enriquecedora experiencia que esperamos sirva de apoyo para cualquier lector que decida tomar como proyecto el hacer un juego… Si bien no es el código más completo y lindo es algo que se aprendió a punta de ensayo y error tanto para ayudarnos a nosotros como a otros.

.

INTRODUCCIÓN

Todos hemos jugados videojuegos en algún momento de nuestras vidas pero no sabemos el trabajo que hay detrás de la pantalla, el cómo se llevan las acciones dentro del videojuego o todo el trabajo que hay detrás de algo que solo vemos el resultado, dentro de este proyecto nos propusimos realizar nuestro propio videojuego debido a nuestra afición por ellos y experimentar de primera mano el trabajo mencionado lo que nos llevó a tener una visión distinta de los videojuegos después de esto, ya que al ver un videojuego ahora parcialmente podemos deducir o imaginar un código que hay detrás del mismo.

Dentro de este escrito nos concentramos en plasmar todos los conocimientos que hemos adquirido de las clases y de investigaciones posteriores de cómo realizar un videojuego por nuestras propias manos.

Manual de Usuario

Para poder llevar a cabo este proyecto y también para poder ejecutarlo como juego , se recomienda al usuario usar:

1.- Python 3.8.10 o superior
2.- Pygame 2.1.2 o superior

usar una versión de pygame inferior podría causar problemas de ejecución que impidan la ejecución completa del código.

Manual de juego

IMPORTANTE
Para abrir el juego en cuestión se debe iniciar el archivo llamado “Menu Inicio.py”
el cual abrirá el menú principal del juego como tal.

Tendrá varios botones visibles (que aún no tiene texto) pero…
el superior es directamente iniciar el juego
el inferior es cierre de juego
y los botones de la parte inferior izquierda de la pantalla tendrán el control de la música del menú principal.

si apreta la tecla a o d:

el personaje acelerará en la dirección siendo **a** la izquierda y **d** la derecha

si apreta la tecla s:

el personaje se agacha como tal

si apreta espacio :

el personaje saltara si aprieta el espacio por 2da vez cuando el personaje empieza a descender:

creará una segunda instancia de salto como doble salto

puedes usar el shift izquierdo para activar el “dash” del personaje

si hace clic en pantalla:

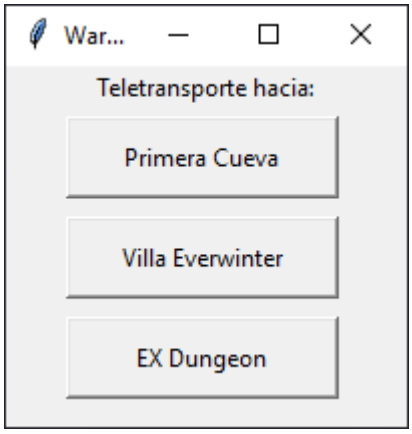
activará el golpe del personaje

también tienes una habilidad especial cuando apretas la tecla e

usarla te gasta 1 punto de maná, que reducirá el contador de veces que puedas usar esta habilidad, si matas un enemigo podrás recuperar un punto de maná

si apreta la tecla f estando en contacto con la lamparita

activará el manejo de evento para activar distintas cosas



si apreta la tecla n después de haber activado el primer nivel de la lámpara

en este caso “primera cueva”:

activará oleadas de slimes cada cierto tiempo
empezará con 1, luego 3, 5 y así sucesivamente

Mantén un ojo en la barra de vida:

ya que los slimes son bastante agresivos. ¡usa tu herramienta de ataque para derrotarlos!

ya que si esta barra de salud llega a 0

el juego se termina...

La experiencia completa para este prototipo de juego en este momento es interactuar con el primer altar(lámpara), activar la primera opción de dungeon , apretar la tecla **n**, e ir derrotando a los slimes que vendrán por oleadas, cada término de oleada deberá ir seguido de un nuevo tecleo de la letra **n** para seguir en la siguiente y así sucesivamente. El objetivo del juego como tal era derrotar una cantidad específica de enemigos slimes (10) para poder liberar la cueva que está en el fondo y que apareciera el enemigo final (un ratón)

Personaje y Enemigo

Siendo la esencia y corazón de los videojuegos, el personaje es un fragmento de código que abarca la mayor parte de variables que pueda estar dentro del mismo código ya que tal interactúa con todos los objetos tanto visibles como invisibles.

Uno de los principales desafíos para programar un personaje dentro de codificación python y pygame es trabajar con las funciones rect que son prácticamente crear rectángulos, los cuales pueden interactuar y colisionar y en base a eso sacar condicionales ya que su colisión devuelven valores booleanos con los que trabajar, el tema en cuestión es que al ser rectángulos las colisiones son más imprecisas ya que no tenemos control completo del contorno del personaje como tal.

En la imagen adjunta podemos ver que el sprite 0 correspondiente al personaje tiene un contorno mayor al del mismo ya que la imagen png trabaja con los pixeles invisibles que tiene el mismo por lo cual es mas fácil trabajar con este tipo de formatos dentro de pygame pero tiene sus limitaciones como se puede ver ya que es impreciso para personajes que no son literalmente rectángulos o cuadrados por lo que una colisión con el enemigo slime…

se producirá antes que visualmente se vea la colisión ya que los píxeles invisibles están en contacto antes que su parte visible.

Si bien ese fue el mayor problema como tal, lo demás fue prácticamente ensayo y error de variables y condiciones más fáciles ya que trabajar con clases, contadores y

condicionales que visualmente se pueden interpretar es más interactivo para aprender a programar

El ejemplo más claro es darle un contador a las imágenes cargadas dentro del programa lo cual genera que en una lista de imágenes , se vayan recorriendo una a una para generar la sensación de movimiento del personaje tanto para simular su respiración, como sus movimientos en general.

Todas estas experiencias sirven y aplican para el mismo Enemigo slime, el cual en parte conserva la esencia del personaje con menos condiciones y menos variables , lo cual fue más fácil de hacer una vez se tuvo comprendido el sistema de clase del personaje ya que luego de esto , usar la herramienta spritecollide de pygame fue clave para trabajar con las colisiones en general de todo el programa, todo en base a ensayo y error para corregir posiciones, movimientos y condiciones de vivir, golpe o morir.

Mapa

Dentro de las principales estructuras que componen a los videojuegos una de estas es el hecho de donde se mueve el personaje o como se sabe tiene que haber algo en cierto lugar para que el personaje interactúe con el medio, debido a esto una las partes del proyecto que nos enfocamos fue en el tema de cómo generar un mapas dentro de pygame, la manera más sencilla de generar que se descubrió es con el tema de una matriz dentro de python pero debido a que esta era poco eficiente debido a que los mapas son muy grandes llenar en matriz de tal magnitud nos llevaría mucho tiempo por lo cual se descartó de manera inmediata, luego de seguir indagando llegamos a unas softwares que con los que se podían generar mapas de manera más fácil y general los para así agilizar la generación de múltiples mapas en corto tiempo, los programas encontrados fueron Tiled, Texture Pack GUI.

Donde estoy programas funcionan de la mano con el otro para así poder llegar a lograr la generación de los mapas, donde Texture Packer GUI se encarga principalmente de crear un hoja de tiles con todas la imagenes a ocupar en el mapa y además de eso genera un archivo json con la estructura de donde se encuentra cada tile dentro de la hoja de tiles. Ejemplo Anexo 1.0

Una vez ya creada la hoja de tiles podemos empezar a ocupar el programa Tiled que nos va a crear el mapa pintando con los tiles para luego exportar el mapa en alguno de los diversos formatos que tiene el programa pero en nuestra situación lo hicimos en un archivo .csv.

Luego de esto nuestro mayor reto fue el hecho de cargar todos estos archivos dentro de pygame para poder así lograr que se pinten los tiles dentro de una superficie se necesita exportar el archivo .csv dentro de una matriz para luego ser leída de mejor manera, debido a que el programa tiles asigna al el número de id de lo que debe pintar en base a la cantidad de imagen que hay y siempre empieza por el 0, dentro del csv el 0 representa a la primera imagen osea “T0” y con el resto de imágenes el número que

hay dentro del archivo csv es el que número que tiene el nombre solo que antes tiene un “T”. Diagrama de flujo Anexo 1.1

Menú

Para generar los diversos menús del juego para que el jugador pudiera interactuar y navegar a través de este lo hicimos con diversos ciclos dentro de funciones en archivos .py para luego llamar a los diversos ciclos fuera del código principal de manera más fácil y limpia, donde principalmente para realizar los menús dentro del juego ocupamos en la mayoría la misma lógica la cual era la de pintar los botones donde se debe de realizar una acción para así darle un acabado visual más bonito y luego usamos las diversas herramientas que aporta pygame, para realizar una acción de tal botón dentro del juego ocupamos una lógica simple la cual se basa en base a la posición del mouse dentro del display y por ejemplo la posición del mouse está en el rango donde esta el boton y además de eso da click entonces tiene que realizar una acción.

Menu Explicación visual

Considerando en base la imagen que puede observar a un costado como nuestra display podemos ver que hay dos colores a los costados del botón donde el color amarrillo representa los limites del boton en el eje X y el de color rojo repsenta los limites del boton dentro del eje Y, ahora referente a lo anteriormente mencionado el mouse dentro del la displey y igual se sabe donde esta en base a una tupla de coordenada (X,Y) por lo cual si es que el mouse se encuentra dentro del rango de el boton en el eje X y eje Y, ademas sumadole un click entonces podemos hacer una interaccion con el videojuego de manera mas amigable y intuitiba para que el usuario pueda disfrutar el videojuego sin mayores complicaciones. Diagrama de flujo Anexo 1.2

Conclusiones

A manera de resumen dentro de este proyecto nos hemos planteado el realizar un videojuego perteneciente al género al mismo que pertenece Mario Bros osea del género plataformas, para poder llegar a un resultado decidimos dividir el proyecto en 3 partes fundamentales mapa, menús, personaje para así poder ver de mejor manera lo que tenemos que hacer para llegar a terminar el proyecto, lo cual se logró de cierta manera pero no en su forma más pulida debido a que lleguemos a la conclusión de que realizar un juego que funciones perfectamente debe de pasar por muchas fases de testing y debug

ANEXOS Y BIBLIOGRAFÍA

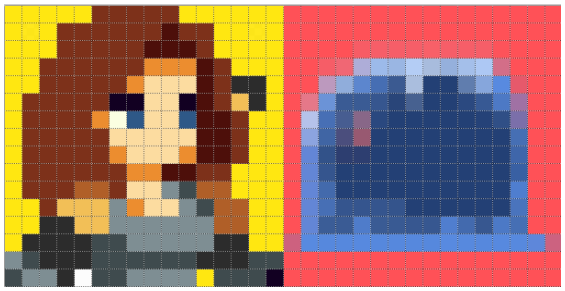
Anexo GitHub

https://github.com/Crisyostin79/Proyecto-Programacion

Anexo 1.0

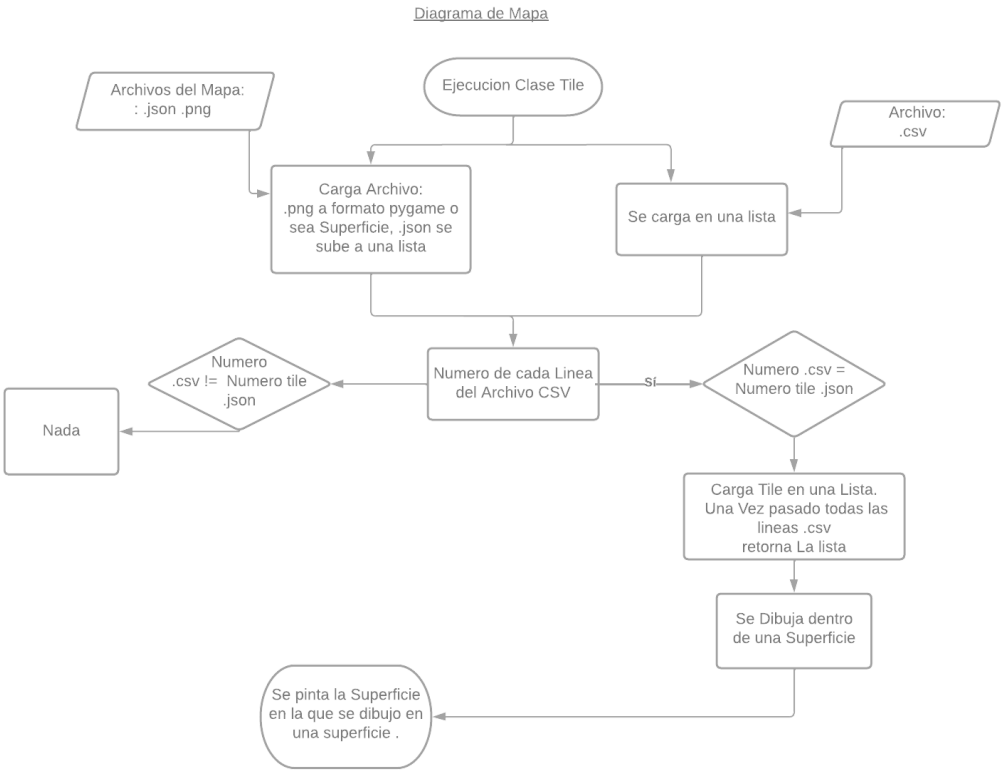

```
{"frames": {

"T0":
{
    "frame": {"x":0,"y":0,"w":16,"h":16},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":16,"h":16},
    "sourceSize": {"w":16,"h":16},
    "pivot": {"x":0.5,"y":0.5}
},
"T1":
{
    "frame": {"x":16,"y":0,"w":16,"h":14},
    "rotated": false,
    "trimmed": false,
    "spriteSourceSize": {"x":0,"y":0,"w":16,"h":14},
    "sourceSize": {"w":16,"h":14},
    "pivot": {"x":0.5,"y":0.5}
```

Aquí se puede ver un ejemplo de la función del Texturepacker GUI donde nos crea un json con la estructura de donde se encuentra de donde se encuentra la imagen, donde x , y representa la posición de la imagen en la hoja de sprites y además w , h que representan el alto y ancho de la misma sin dejar de lado el hecho que se guarda con el nombre la imagen o en este caso ”objeto”.

Anexo 1.1 Diagrama de Mapa



Anexo 1.2 Diagrama Menú