1. Bucle principal
    a. En qué consiste
    b. Como funciona
2. Empezar hablando sobre las mecánicas del juego y como se implementan
    a. Controles en mapa
    b. Controles en nivel
    c. Colisiones
    d. Máquina de estados
    e. Localizaciones y registro de localizaciones
3. Tiled y como se aplican los dibujos al juego
    a. Explicar Tiled
    b. Animación mapa
    c. Animaciones nivel
    d. Json de los sprites
4. Ventana de inicio de sesión

https://prezi.com/view/KomIlPvQzGjyROohPbXO/

http://bowiesgame.epizy.com/

https://app.infinityfree.net/accounts/epiz_28835904

http://185.27.134.10/db_structure.php?db=epiz_28835904_bowiesgame

Les presentaré mi proyecto de fin de grado (ENSEÑA EL JUEGO). Por muy tonto que parezca el juego, para mi ha sido todo un logro y una aventura desarrollar este juego. (SOBRE EL JUEGO). No inventa nada nuevo sobre el género, de hecho, una unas mecánicas simples y mas que refinadas en la industria (SCROLL).

Sobre el juego

Es un juego de Scroll lateral en 2 dimensiones. Existe un mapa que contiene todos los niveles accesibles del juego. Este mapa tiene una vista cenital para que el usuario pueda ver con facilidad todo mapa moviéndose hacia arriba, abajo, izquierda y derecha con las teclas WASD del teclado (MIENTRAS MOSTRAR JUEGO SIMULTÁNEAMENTE). Una vez dentro del nivel, los controles cambiarán, nos moveremos solo de izquierda a derecha, caeremos y podremos saltar con la barra espaciadora. Para que nos hagamos una idea clara, este tipo de mecánicas son las de los clásicos Mario Bros de la NES o cualquier consola de Nintendo.

Puntos clave

Los puntos clave del proyecto son, el bucle principal, que es lo que realmente hace que el juego se mueva y funcione, las mecánicas, que ya comentamos un poco por encima, que son el movimiento, el salto, etc, las localizaciones para acceder a los niveles, comentaremos también el apartado grafico y como se crearon los niveles y el mapa, y por último la ventana de inicio de sesión que aquí os puedo mostrar (ENSEÑAR LA VENTANA DE INICIO DE SESIÓN)

BUCLE PRINCIPAL

Dentro de lo que cabe es relativamente simple. Queremos que el juego se vea fluido y que no de tirones, para lo cual necesitamos que se mueva a 60 frames por segundo, o lo que viene a ser lo mismo, que el juego se dibuje y actualice 60 veces por segundo.

Centrémonos en esta parte del código, en la función "**iterar**". Como es necesario que el juego siempre se esté ejecutando, esperando a que el usuario realice una acción, como moverse o abrir la ventana de inicio de sesión, es necesario que haya un bucle que se repita hasta el infinito.

Para que la ejecución se capture y de vueltas hasta el infinito sin salirse nunca y a 60 veces por segundo, gracias a las tecnologías de HTML5, esto no es muy difícil de conseguir, porque el **navegador lo hará por nosotros**. Lo primero es acceder la id de ejecución y lo iniciamos con el objeto **`window.requestAnimationFrame().`** Este informa al navegador que se quiere realizar una animación y solicita que el navegador **programe el repintado de la ventana** para el próximo ciclo de animación. Llamando de nuevo a la función "iterar", se consigue hacer un **Callback**. Cuando se ejecuta esa parte del código, el valor que devuelve es el tiempo medido en milisegundos desde su ejecución, e inyecta ese valor en **`registroTemporal.`**

MECÁNICAS

En el juego existen dos mecánicas muy básicas y fáciles de identificar, pero con sus particularidades, que son el movimiento del personaje dependiendo de su estado.

Necesitamos que el juego sepa reconocer cuando está en el mapa y cuando está en el nivel, lo cual hace que necesitemos una **máquina de estados** que vaya variando entre todos ellos. También necesita una función actualizar y dibujar para que todo esto se vaya aplicando a medida que vamos jugando.

Cuando estamos en el **mapamundi**, el personaje se moverá de izquierda a derecha, arriba y abajo para poder desplazarse por todo el mapa y así llegar a acceder a las localizaciones para entrar al nivel.

Una vez estemos dentro del nivel, no podremos subir y bajar como tal, pero si seguiremos pudiendo ir de izquierda a derecha. En los niveles hay unas plataformas que debemos subir para poder finalizar la fase, para ello usaremos el salto con la barra espaciadora, pero podremos caer al suelo, ya que se implementa una función que simula la gravedad.

Las mecánicas del movimiento son relativamente sencillas: cuando el jugador no encuentra una colisión arriba, y en el teclado estamos pulsando la tecla arriba, la velocidad en el eje Y aumentan, así le estamos indicando que nos movemos hacia arriba, en el caso de estar en el mapamundi yendo hacia arriba.

Para el nivel es distinto, porque tenemos que aplicar ciertas condiciones. Tampoco quiero pararme demasiado en este apartado, ya que es todo muy técnico y un poquito tedioso de explicar todo .

- Si el salto está bloqueado, si nos estamos chocando hacia abajo y si no hemos pulsado la tecla de saltar, lo que haremos será movernos sobre el mapa.
- Si no tenemos el salto bloqueado y estamos pulsando la tecla de saltar tenemos la opción de saltar y automáticamente, lo bloquearemos para no poder volver a saltar.
- Si no tenemos una colisión arriba y estamos subiendo, lo que haremos será reducir los frames aéreos y la velocidad del eje Y será igual a 1 multiplicado por la velocidad de movimiento más los frames aéreos. Lo que conseguimos con esto es poder subir en el salto rápidamente, pero a medida que va llegando a su altura máxima, la velocidad incrementará más lentamente hasta llegar al punto de parar y volver a caer hacia abajo. Para esto último, debemos de comprobar si los frames aéreo son menores o iguales a 0
- Si los frames aéreos son menores o iguales a 0, os sea, no podemos subir más, tendremos que indicar que ya no estamos subiendo y que los frames aéreos han llegado a su máximo, o también se puede decir que reiniciamos el contador.
- Si no estamos colisionando con nada abajo, y no estamos subiendo, o sea, estamos en el aire (cayendo), tenemos que hacer que la velocidad Y se acelere mientras caemos sin que haya decimales, para que sea constante. Si la velocidad de la caída es menor que la velocidad terminal, le iremos sumando 0.3 a la caída, que es un valor que hace que la caída se vea lo más natural posible.
- Para movernos a los lados comprobamos si no tenemos una colisión a la izquierda, por ejemplo, y estemos pulsando la tecla izquierda hacemos que la velocidad X sea 1 * la velocidad de movimiento, y lo mismo para la derecha.

También es necesario una función que nos compruebe si estamos colisionando con algún elemento o con el borde exterior del juego, para luego poder aplicarlo al resto de funciones del programa, una función dirigir, que funcionará de una forma u otra dependiendo del estado del juego, si está en el mapamundi o no, bloqueando así su movimiento vertical, y por ultimo una pequeña función donde animaremos a nuestro personaje. Esta no es muy relevante, pero siempre de agradecer un ligero movimiento amigable de nuestro personaje para darle un poco de carisma.

LOCALIZACIONES

Hablemos ahora de las localizaciones. Las localizaciones funcionan de una forma muy similar a las colisiones, ya que una parte de su función es detectar si una hitbox, la del personaje, entra en contacto con la de la localización, con la diferencia de que, en vez de hacer que este se detenga, se pueda atravesar y a demás acceder a un nivel específico. Hay varias localizaciones repartidas por el mapa, todas tienen nombre y pero solo 3 de ellas contienen un nivel

diseñado. También hay que decir que dentro de los niveles existen localizaciones que nos indican el final del nivel. Para acceder al nivel pulsamos la tecla “E” y para salir la tecla “X”.

Para finalizar con este apartado, habréis podido observar como aparecen unas ventanas emergentes sobre cada nivel. A esta ventana se la llama popup el cual nos indica el nombre del nivel al que accederemos. Esta función es totalmente irrelevante, no tiene una función importante en el juego, pero queda muy bien.

GRÁFICOS

Una herramienta indispensable es la de “seleccionar objetos”. Su función es la de generar un área delimitada por unas coordenadas registradas en un fichero *.json*, un identificador y un color para diferenciarlo todo visualmente.

Aquí podemos observar cómo, al poner visibles las capas de objetos, se nos muestra de forma visual dos tipos de áreas distintas, una verde y una roja. Se puede intuir a simple vista la función de cada tipo de área:

- **Colisiones**: los cuadrador rojos delimitan las zonas por las que nuestro personaje no podrá caminar, a esto se le denomina “hitbox”. Se les suele aplicar a zonas de agua no navegable, arboles, montañas, delimitando el mapa y cualquier zona a la que no deseemos que el jugador acceda .
- **Localizaciones**: los cuadrados verdes marcan las ubicaciones a las que el jugador puede acceder para entrar a un nivel. Estas tienen un nombre, que hace de identificador. Al pulsar una tecla especifica, el usuario accederá al nivel en cuestión.

Para finalizar, debemos exportar el fichero con dos formatos distintos: *json* y *png*. En el *json* podremos encontrar las distintas capas creadas, con las coordenadas en las que se crearon, su tamaño, su nombre y varias características del mismo. A demás de todo esto, hay muchos mas datos, como por ejemplos los *tilesets,* que son los datos propios del mapa: nombre, ruta, alto, anto, etc.

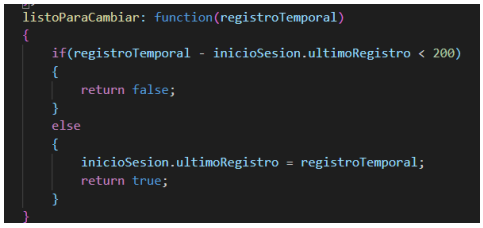
INICIO DE SESIÓN

Invocaremos un rectángulo con unos colores y diseño totalmente personalizable donde podremos encontrar las opciones de iniciar sesión y guardar partida.

Comenzamos creando el div y las etiquetas HTML necesarias para introducir todos los datos en el fichero **`index`**.

Aquí podremos reutilizar el fichero de **popup**, ya que usa funciones similares. Añadimos **`ultimoRegistro: 0`**, el uso que tendrá será para saber cuándo se abrió para evitar ciertas complicaciones a la hora de abrir y cerrar la ventana. Sustituimos en la función mostrar el ancho y el texto por un registro temporal que lo alimentamos en el bucle principal cada vez que actualizamos y dibujamos el juego. Fijamos el ancho, creamos una id para aplicarle los estilos a la ventana, y aplicamos los estilos que deseemos.

La función llamada **listoParaCambiar**, que usará el registro temporal como parámetro. Su finalidad será comprobar saber hace cuanto tiempo pulsamos la tecla para abrir la ventana para poder aplicarle un "delay" a la hora de abrirlo. Esto se hace para que, si el usuario le hace click muchas veces seguidas o mantiene el botón pulsado, no esté abriéndose y cerrándose continuamente; tendrá un mínimo retraso de 200 milisegundos.



Una vez hecho esto, vamos a la parte del código donde le indicamos que, si la ventana ya está visible, no podamos volver a abrirla, y le añadimos que, si la ventana no está lista para cambiar y lo alimentamos con el registro temporal, le hacemos un return. Así nos podremos saltar la función. Lo mismo se lo aplicamos a la función **ocultar**.

Lo que nos quedaría sería borrar la línea del **innerHTML** y dirigirnos al fichero **controlesTeclado** y asignarle una tecla la cual tendrá la función de hace que aparezca y desaparezca nuestra ventana de inicio de sesión[ç].

Para finalizar, nos dirigimos a E**stadoMapamundi** hay que añadir un poco de código justo debajo de donde aplicamos el popup, incluimos lo siguiente:

```
if(teclado.teclaPulsada(controlesTeclado.inicioSesion) && !inicioSesion.visible)
{
    inicioSesion.mostrar(100,100, registroTemporal);
}
if(teclado.teclaPulsada(controlesTeclado.inicioSesion) && inicioSesion.visible)
{
    inicioSesion.ocultar(registroTemporal);
}
```

Es simple, lo que hará es que aparezca y desaparezca nuestra ventana de inicio de sesión usando la función **mostrar** y **ocultar**. En el **mostrar** es necesario también una X y una Y, la cual añadimos justo antes del registro temporal.