Pseudocódigo de “MultiSimpson”

Haríamos un juego basado en guerra de clones, con la finalidad de poder hacer aparecer más de un personaje igual basado en los Simpson.

La creación del mapa se basará en un fondo 2D y sprites de los Simpson. En el mismo, los defensores se colocarán a la derecha de la pantalla, y los atacantes irán apareciendo del lado izquierdo. El objetivo es evitar el paso de TODOS los enemigos hacia el lado derecho.

Con cada ronda se incrementará la cantidad de enemigos difíciles que aparecerán. Utilizaremos el mismo formato que el de las probabilidades de dropear un objeto mágico cuando muere un enemigo.

Los defensores utilizaran los pixeles como forma de desplazamiento por el mapa.

En el juego habrá niveles, y cada nivel estará formado por tres rondas.

La clase mapa:

(2) puedoAvanzar(posicion)_ se encarga de verificar (con el mapa) si el personaje puede avanzar los siguientes pixeles (siguiente).

avanzar(posicion)_ se encarga de hacer que el personaje avance a las nuevas coordenadas (en pixeles).

(1) hayEnemigo()_ se encarga de ver si hay un personaje del equipo contrario en el radio de acción del personaje. Si hay, se encarga de devolver el ***otro enemigo*** más cercano a este primer personaje.

La clase personaje:

puedeAtacar()_ se encarga de preguntarle al mapa si hay un enemigo para atacar.(1)

atacar(***otro enemigo***)_ luego de verificar que se haya un personaje del equipo contrario en el radio de acción del personaje, el personaje "ataca" al otro; se efectúa el daño correspondiente sobre el segundo personaje de la ecuación.

Setters y getters para obtener el estado y modificarlo.

La clase defensor:

activarBonus(bonus)_ y se va a encargar de modificar los atributos para hacer más daño/ velocidad/ etc.
desactivarBonus(bonus)_ en caso de tener un bonus activado, devuelve todos los atributos a su estado inicial.

La clase enemigo:

avanzar()_ Pedir nuestra posición. Una vez que la tenemos le preguntamos al mapa si podemos avanzar (2) mapa.puedoAvanzar(this? -- o directamente la label?) si retorna un valor de verdad mapa.avanzar(this? -- o directamente la label?)

Los objetos pueden ser de dos tipos, con vida (como podrían ser barricadas o piedras) o por tiempo, los cuales tienen vida infinita pero desaparecen después de un tiempo (como los objetos mágicos, lagos o fuego). Entre las hordas, pueden aparecer en el mapa objetos propios del mismo para ayudar al jugador.
Al ser destruidos, algunos de los enemigos dejarán un premio, el cual el jugador deberá "tocar" (click) para obtener.
Estos premios pueden ser de dos tipos:
- *Magia temporal: Este tipo de premios otorga efectos temporales a todas o a algunas de las unidades del jugador pudiendo por ejemplo duplicar la fuerza. Uno de los efectos deberá ser una suerte de "campo de protección" que destruya al siguiente enemigo que lo toque.*
- *Objetos preciosos: Este tipo de premios le confiere al jugador objetos especiales los cuales puede colocar en el mapa, pudiendo estos ser bombas o barricadas. Hay que tener cuidado con las bombas porque estas afectan a todos, tanto enemigos como personajes del jugador.*

Cada tanto, algún enemigo aparecerá con magia aplicada sobre el.

Pero entonces un elemento puede ser por ej. un objeto temporal y un premio de clase magia al mismo tiempo?? SIIII

La clase objeto:

La clase precioso:

La clase magia:

La clase temporal:

La clase conVida:

La clase juego:

Tendrá un mapa, arreglos?, de cada tipo de dato (objetos del juego - GameObject) que utilizará el juego (Enemigos, Defensores, Objetos, etc.) lo cual facilitará la automatización de una elección en ciertas ocasiones.

Dentro de la misma podríamos tener un arreglo de niveles. ¿? Y la clase niveles a su vez un arreglo de rondas. ¿?**

Nota: En caso de utilizar random con probabilidades, podemos limitar el rango para que solo tome los primeros enemigos en la ronda uno, por ejemplo.

La clase nivel: (coleccion de rondas)

Tendrá atributos que modelen las rondas.

Dependiendo el nivel, se llevará a cabo cierta asignación de rondas al nivel, y el nivel se las hará conocer al juego.

La clase ronda:

Las rondas van a tener una cierta cantidad de enemigos que se deberán eliminar para completar la misma. La cantidad de enemigos va determinarse por el número de ronda y nivel en el que se encuentra el jugador. Por ejemplo:

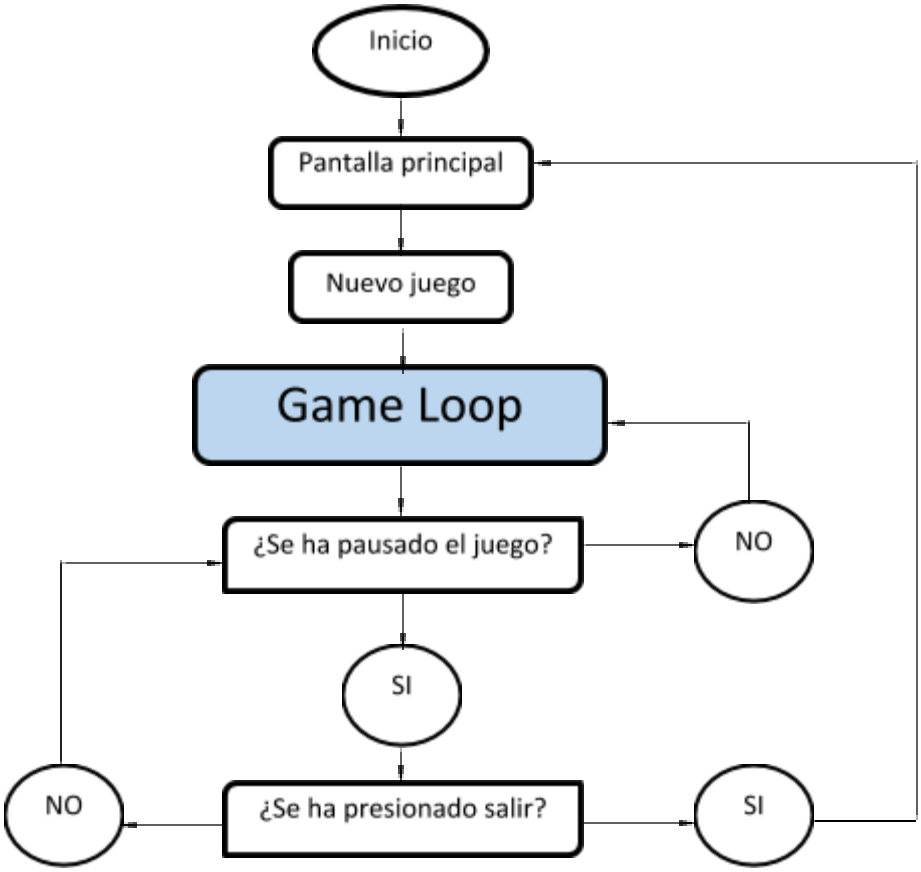
Nivel 1 - ronda 1: 10 enemigos

Nivel 1 - ronda 2: 15 enemigos

Nivel 1 - ronda 3: 20 enemigos

Y así.

Los enemigos que aparecerán serán especificados por esta clase de forma aleatoria, también según la ronda y el nivel actual. **Asimismo, las apariciones de enemigos tendrán un tiempo de cooldown (entre apariciones) -- no pueden aparecer todos juntos -- cómo los separamos?**. A mayor nivel y ronda más avanzada, aumentarán las probabilidades de que aparezca un enemigo más raro/de más jerarquía.


Inicio
Pantalla principal
Nuevo juego
Game Loop
¿Se ha pausado el juego?
NO
SI
NO
¿Se ha presionado salir?
SI