**Guía Scrolling Shooter**

1. Una variable sirve para almacenar información en los dispositivos y así utilizarla después, debe ser nombrada para poder referirse a ellas en ciertos programas y funciones.
2. El Scope de una variable es el alcance que esta tiene, es decir, el área donde esta puede ser utilizada.
3. Los tipos de variables existentes dentro de Game Maker son:
   1. Instancia: Están definidas dentro de una instancia, Son únicas para esta instancia y pueden usarse en cualquier evento o función dentro de esta.
   2. Local: Son variables temporales que solo son accesibles dentro del código que estamos ejecutando en ese momento.
   3. Global: Son aquellas que pertenecen al juego y no a alguna instancia. Al poner una variable el prefijo global, esta permite que desde cualquier objeto se puede llamar y modificar su valor.
   4. Construcción dentro de variables o variables del sistema: Son variables especiales que ya vienen creadas dentro de Game Maker. Estas variables pueden ser instancias o globales en scope.
4. Las variables en Game Maker pueden ser de ámbito local, cuando ha acabado de ejecutarse el evento o el script donde están escritas se eliminan para que no ocupen memoria, pueden ser de ámbito global, es decir, da igual desde dónde se llame, cualquier objeto o script, solo existirá una variable y su valor será el mismo en todos lados o de ámbito instancia, o sea donde se crea esa variable y se le da ese valor, es visible dentro del objeto.
5. Usaría la variable instancia ya que esta permite acceder a los eventos que están relacionados con una misma instancia.
6. Usaría una variable local, porque esta permite acceder a la información del código que se está utilizando en el momento.
7. Usaría la variable global, a que me permite acceder a información de una variable desde cualquier instancia
8. Score, healt, lives, global.total\_enemies, globalvar speed.
9. Potions, life, strength, armour, agility
10. No, ya que game maker distingue entre mayúsculas y minúsculas por lo que sería dos variables diferentes.
11. A) se crea en la parte superior exactamente en el centro del myplane. B) La opción relative significa que las coordenadas que se dan es relativa al objeto y no al room.
12. Personalización.
13. Para que las islas no interrumpan el vuelo ni las balas así no tiene nada con que colisionar si no que son un objeto de “decoración”.
14. Se hace para que las islas no aparezcan en el mismo lugar y no se vea como un mapa lineal.
15. Ya que las islas inicialmente están posicionadas por el usuario.
16. Lo que hace es que randomiza la instancia descrita con esta variable haciendo que haga aleatoriamente un evento
17. Porque al ser una prueba puede verse como una falla en el juego
18. Lo recomendable es que se llame al inicio del juego
19. Se hace este código para hacerle entender a Game Maker que el juego tiene barreras, y lo explicamos que si el avión está a 40 pixeles del borde no pueda saltar.
20. La instancia de la bala se crea siempre relativamente al avión y al poner -16 pixeles en “y” se crea centrado en la parte superior.
21. Para que el objeto se valla destruyendo a medida de que salga del mapa y así no ir consumiendo memoria.
22. Se logra gracias a una alarma que se pone a 15 steps por lo que son 2 veces por segundo.
23. Se tiene que poner la alarma a 10 steps para que así se logre 3 veces por segundo.
24. Suena la explosión, se crea el objeto explosión, se destruye la instancia (self), y se disminuye la vida del player en -30 relativamente.
25. Pone alarmas en los enemigos para que así los aviones sean más amontonados y se vaya aumentando el nivel de dificultad.
26. Es un tipo de contador que sirve para lograr unas funciones en un determinado tiempo.
27. El subimage es el sprite que se quiere utilizar, si un gif animado tiene 15 imágenes, entonces si uno coloca 3 en subimage escoge la 3 imagen del gif.
28. Lo que hace la línea es que coloca una probabilidad de 1 en x en este caso 30, entonces funciona como la probabilidad de sacar un número en un dado de 30 lados y si la probabilidad se hace true entonces se dispara la bala.
29. Dice que si el personaje existe en la pantalla las balas se disparan hacia el si no se dispara con velocidad vertical hacia abajo.
30. Se quitó la acción y no varía en la jugabilidad del juego pero lógicamente aunque el personaje no esté en la pantalla del view los aviones deben estar disparando hacia la dirección de uno.
31. Si no se coloca las balas generarían desgaste de ram y eventualmente haría que se el juego se bloquee o crashee.
32. Dice que al hundir espacio, si la variable de disparar es verdadera (1) y si el puntaje es mayor que 400 se crean dos balas y si es mayor que mil se crea otra si no se cumple entonces se crea solo una bala y se aplica la variable de que no puede disparar.
33. Es el área de cámara que el jugador logra ver en el juego.
34. Porque si se coloca algún movimiento el fondo comenzara a moverse infinitamente.
35. -2 porque se trabaja con pixeles y si se coloca -2 significa que se le restan esos pixeles a la variable y por lo que va hacia arriba
36. Se le coloca una velocidad al view, se mueve a la velocidad deseada por el jugador recomendablemente que sea de la misma velocidad del avión para que no se logren malos efectos visuales.
37. Se le coloca un mando que dice que si está a un números de pixeles del borde de la pantalla salte hacia el lugar donde se encontraba por lo que parece que hubiera un límite invisible.
38. Porque se coloca la función outside room que lo que hace es controlar lo de afuera y si la bala se vuelve verdadera en esta función entonces se destruye.
39. Se les coloca una velocidad vertical si van de arriba hacia abajo una velocidad positiva pero si van de abajo hacia arriba se les colocara una velocidad negativa.
40. La view in room es el área de cámara en el cual se mirara el mapa y a los jugadores, en cambio la port on screen es el área en el cual se trabajara el juego entonces si la view es mayor que la port, la view se adaptara a la resolución de la port.
41. Personalización.
42. Personalización