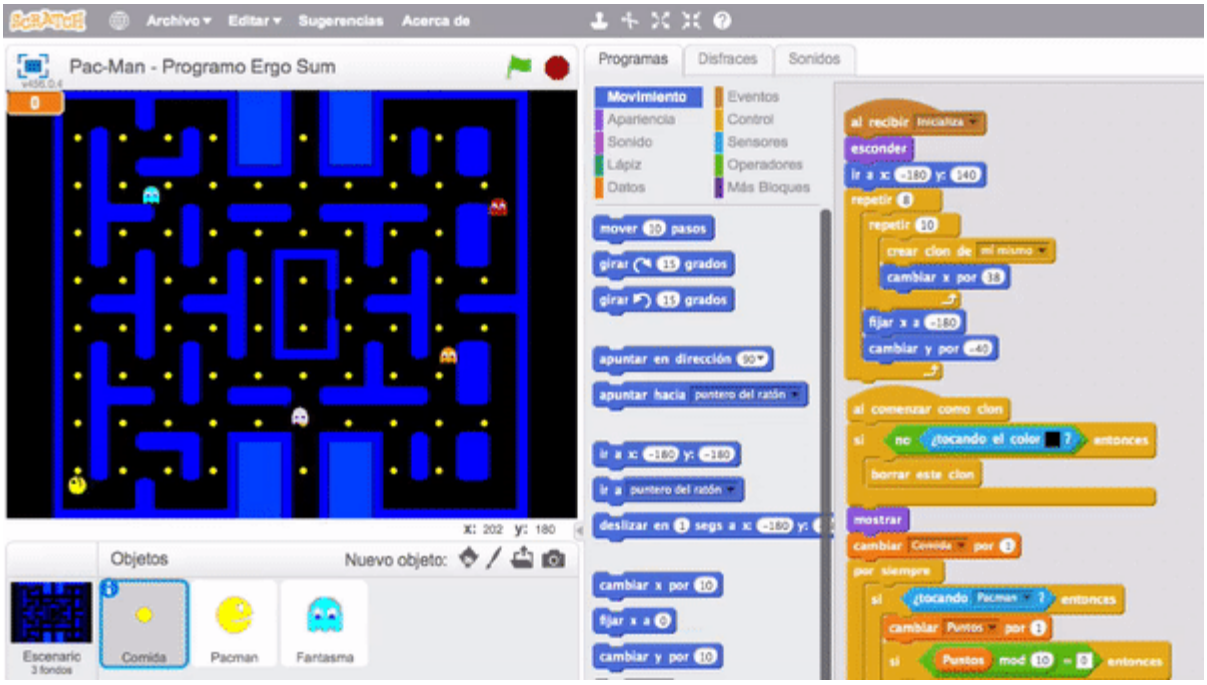
Videojuego de Pac Man programado con Scratch

El Juego

En este tutorial con Scratch 2.0 te explicamos cómo programar el videojuego de Pac-Man. El videojuego consiste en mover al personaje principal Pac-Man para coger la comida esquivando a los fantasmas. Los fantasmas se mueven de forma aleatoria ya que están programados mediante un algoritmo que busca a Pac-Man. El juego termina cuando obtenemos toda la comida.



Reinventa, programa y comparte

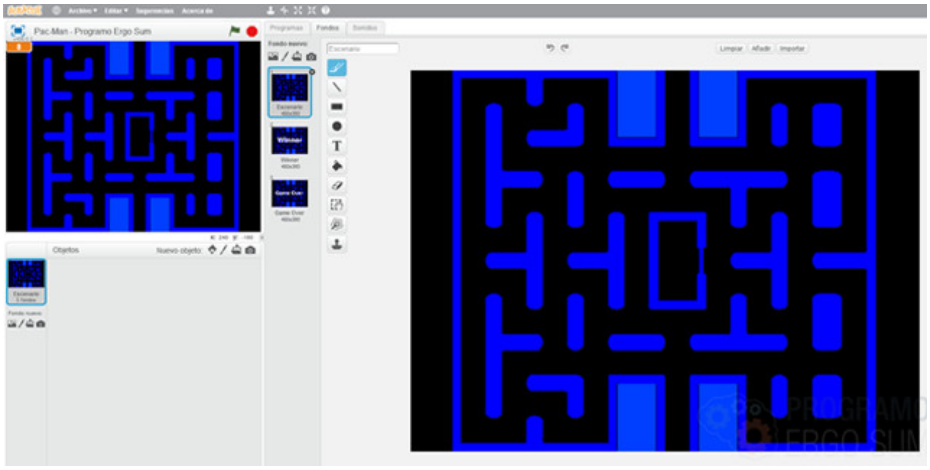
Antes de continuar con las lecciones de este curso de programación con Scratch te recomendamos seguir los siguientes pasos para reinventar y obtener todas las imágenes utilizadas en el videojuego gratis.

Entrar en la Clase y reinventar **Pac Man (base)** para obtener todas las imágenes.
Programa el videojuego siguiendo los pasos de las siguientes lecciones.
Una vez finalizado el juego, comparte el proyecto y si está entre los mejores aparecerá en la sección Mejores proyectos.

Escenario del juego

A la hora de programar videojuegos es muy importante utilizar una buena base de diseño. Normalmente se suelen utilizar plantillas o cuadrículas ya que los personajes se mueven una determinada cantidad de píxeles o pasos.

Para este videojuego se ha creado una cuadrícula para que el personaje principal y los fantasmas puedan desplazarse libremente siguiendo el recorrido y sin atravesar las paredes del videojuego.



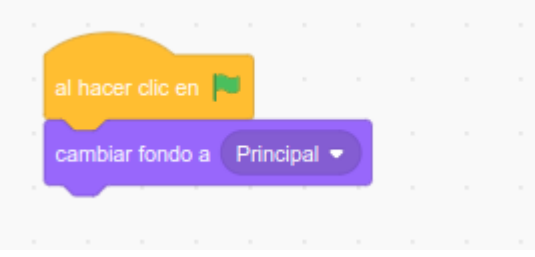
Programación del escenario

En cuanto a la programación en el escenario, aprovecharemos ahí para crear los eventos de "Comienza juego", "Game Over" y "Winner", los cuales cambiarán al fondo correspondiente en cada caso.

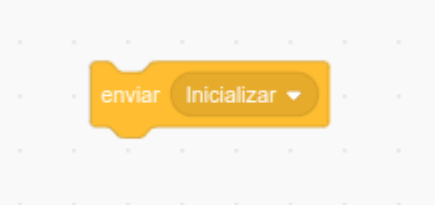
Si hablamos de cambiar escenarios, nos referimos a cambiar el disfraz del mismo, primero debemos crear cada disfraz y lo haremos duplicando el principal pero, colocando un texto en el centro que ponga, un escenario con “Winer” y otro escenario con “Game Over”. A cada disfraz le pondremos su nombre correspondiente, “Principal”, “Winer” y “Game Over”.

El flujo de entrada principal del videojuego es el escenario, ya que en el mismo activaremos cuando aparecen los fantasmas, la comida, etc.

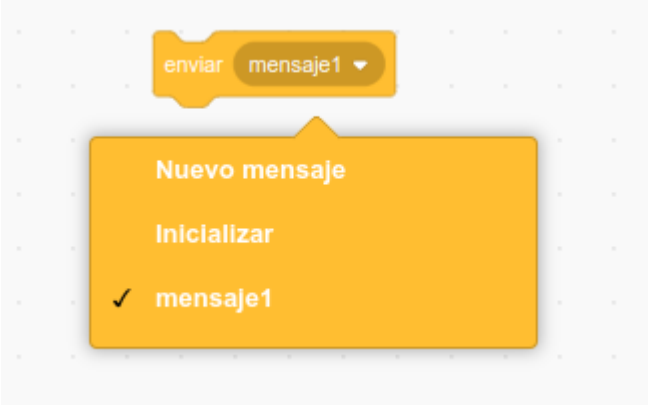
Al presionar la bandera verde debemos obligar a que se muestre el primer disfraz del escenario, es decir, el disfraz “Principal”



A continuación deberemos inicializar dos funciones “Inicializar” y “Comenzar juego”. Esto lo haremos a través de una ficha de la categoría Eventos.

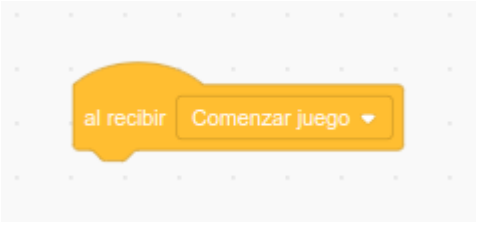


En realidad no la encontrarás así, sino que, en vez de Inicializar, pondrá “mensaje1” y tu deberás cambiarla a través de la opción Nuevo Mensaje del desplegable.

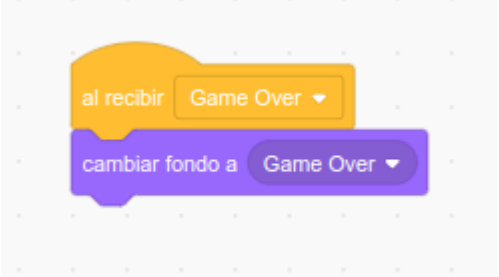


Recuerda que tienes que añadir otra más con el mensaje “Comenzar Juego” y, entre las dos, situar una pausa o espera, esto hará que de tiempo a que la comida, los fantasmas y nuestro PacMan puedan aparecer correctamente en el juego, bastará con 3 segundos, aunque lo puedes variar a tu gusto.

Sin salirnos del escenario, creamos tres órdenes para lanzar programas diferentes. Lo haremos con la ficha de Eventos “Al recibir…”



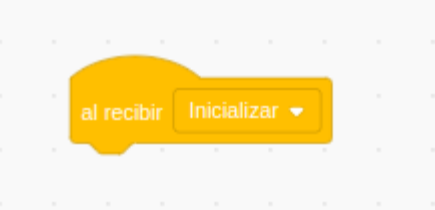
Sólo te aparecen dos opciones, “Comenzar Juego” y “Inicializar”, en principio solo cogeremos la primera pero, crearemos directamente otras dos con la opción de Nuevo mensaje, una que sea “Winner” y otra “Game Over”, que programaremos con una ficha de Apariencia que cambie al disfraz correspondiente del fondo en cuestión.



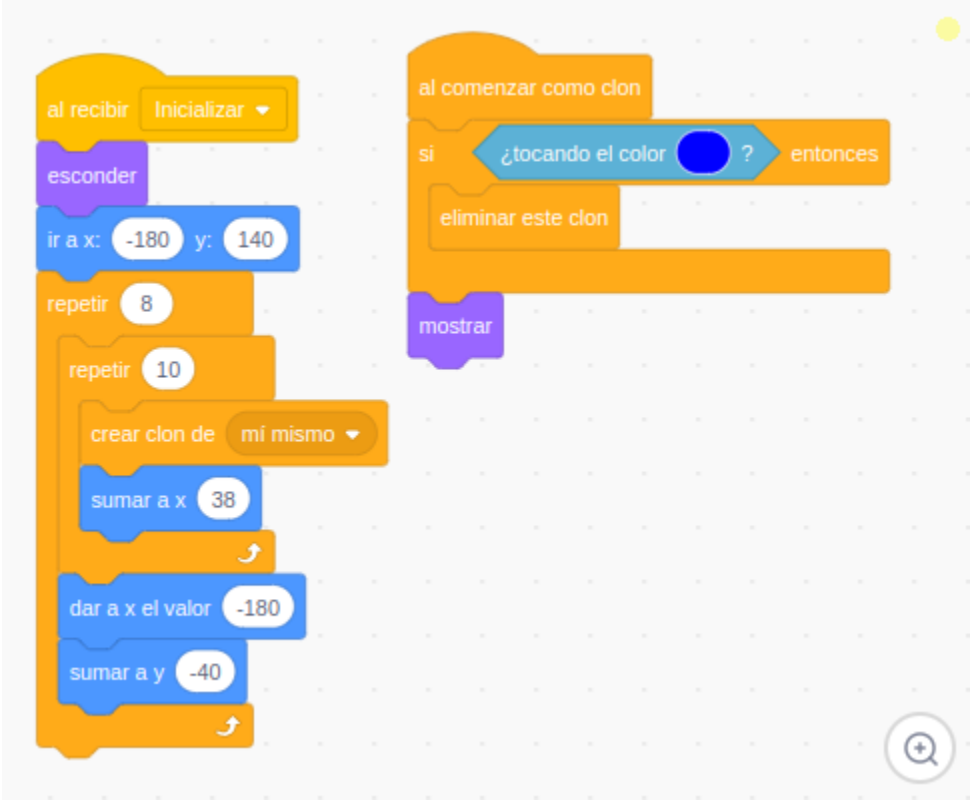
Bucle anidado para la comida

Un bucle anidado no es otra cosa que un bucle dentro de otro.

El primer paso será crear el objeto comida mediante la herramienta de dibujo de Scratch. A continuación utilizamos el evento de "Inicializar" para crear la matriz de comida.

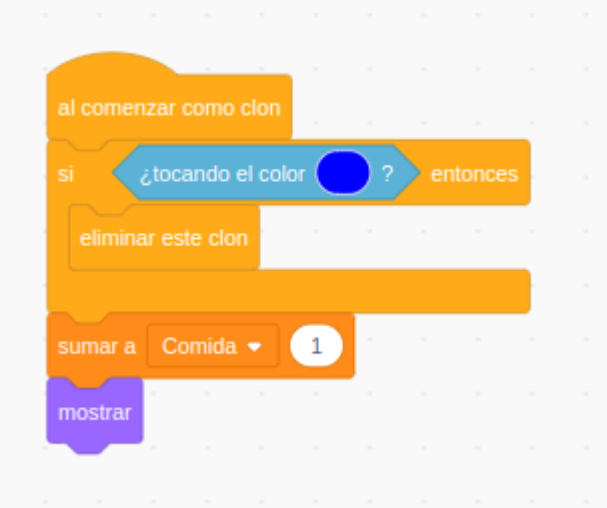


Lo primero será esconder el sprite de la comida, por si la partida anterior perdimos y se quedan elementos por comer. Después situaremos el sprite en un punto en la esquina superior izquierda. A partir de aquí se crea el bucle anidado que creará clones del sprite Comida a lo largo del escenario, como rellenando una cuadrícula. También eliminaremos los clones que coincida con las zonas azules o paredes. Por complejidad, te facilito el código pero, analiza bien cómo está estructurado.



Se puede observar que utilizamos bucles anidados (un bucle dentro de otro) y por cada iteración del bucle interno se crea un clon de comida. Recuerda que como siempre en todos los tutoriales utilizamos buenas prácticas de programación.

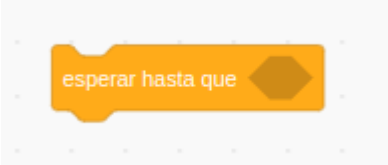
Aún no hemos terminado de programar nuestro sprite Comida. Hay que saber que ganaremos el juego si llegamos a comernos toda la comida, por ello tendremos que saber cuantos puntos de comida aparecen cada vez que se Inicializa el juego. Lo haremos creando una variable llamada “Comida” y añadiendo la ficha de sumar un valor a la misma cada vez que se crea un clon.



Una vez hecho esto, arriba a la izquierda me saldrá la casilla de la variable Comida (luego la ocultaremos) y podremos ver, tras presionar bandera verde, el número de puntitos de comida que se han creado (recuerda que los que aparecían tocando el color azul, desaparecen). Por lo tanto, en el juego, ganaremos cuando nos los comamos todos, ¿y como sabremos que ocurre esto? Pues creando una variable nueva llamada puntos, que tendrá un valor inicial de cero pero, que irá sumando 1 punto cada vez que el personaje principal se coma un puntito de comida, algo que programaremos más adelante.En definitiva, ganaremos la partida cuando Puntos=Comida.

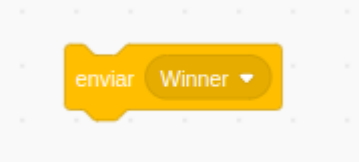
Para realizar esto último, volvemos a programar sobre el escenario y nos vamos a esa ficha que habíamos dejado pendiente, ¿recuerdas?

Bajo esta ficha colocaremos una condición, por lo tanto iremos a la categoría Control y elegiremos la siguiente ficha:



Incorporando dentro de la ficha, otra de la categoría Operadores que compares si el valor de Puntos es igual al valor Comida.

Y si se cumple, tendremos que llamar a la función Winner.

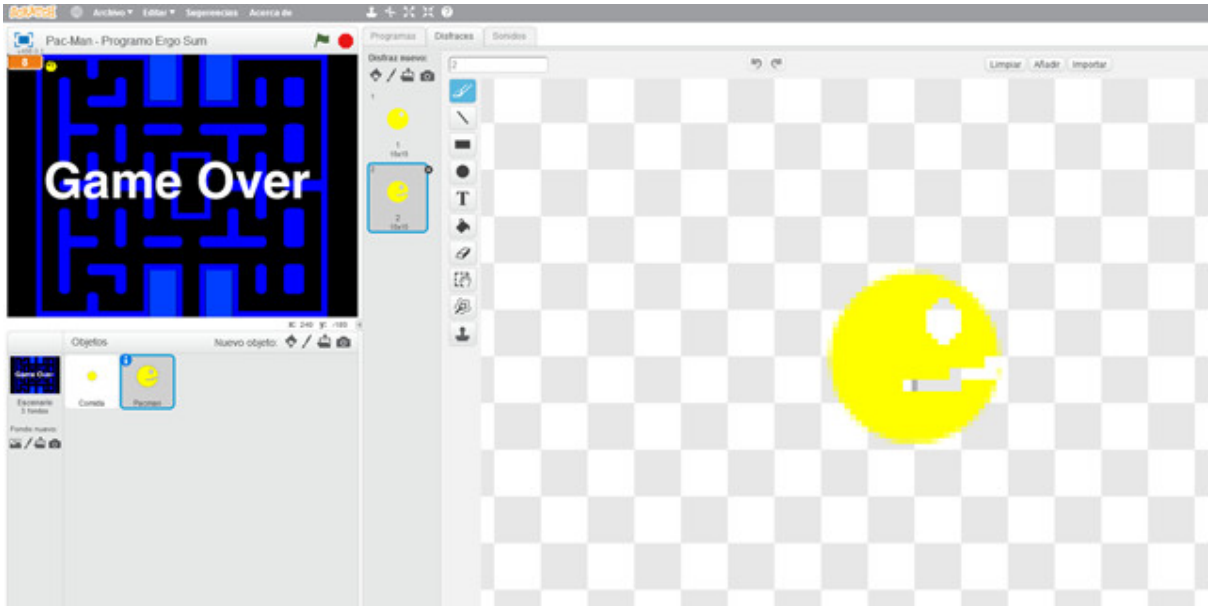


Esta última ficha, recuerda, lo que ejecutará el cambio de disfraz Winner.

Personaje principal

Ahora vamos a ocuparnos del personaje principal, PacMan, El primer paso será añadir el objeto con sus respectivos disfraces para realizar el efecto de abrir y cerrar la boca e inicializar sus atributos por defecto como coordenadas, tamaño, etc.

En este caso nos ayudamos del editor de dibujo de Scratch para crear a Pac-Man pero puedes utilizar otros diseños.



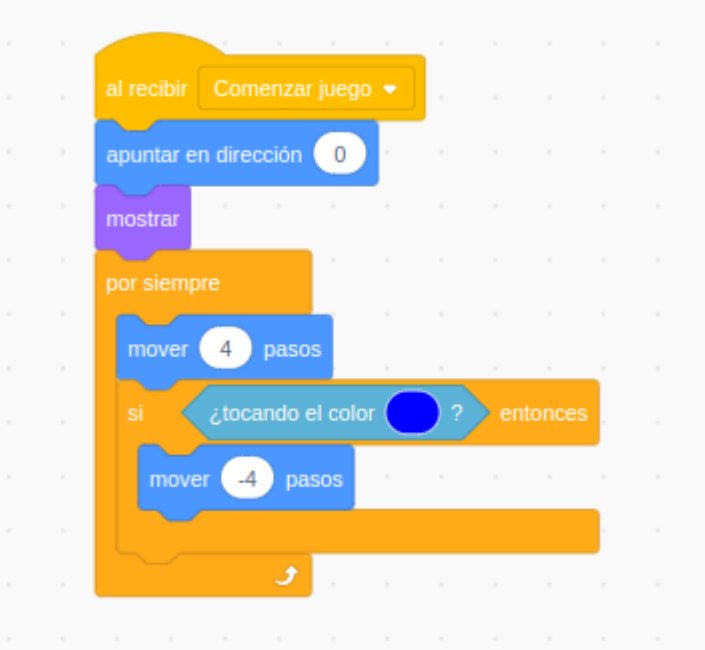
Puedes crear tu PacMan como desees pero, ya tienes uno creado en el programa base.

Programación de Pac-Man

Una vez creado nuestro PacMan vamos a programar como se moverá en nuestro juego. En realidad se moverá sólo automáticamente y nosotros solo nos ocuparemos de cambiar su dirección, arriba, abajo, izquierda o derecha. Para ello optamos por utilizar condiciones en los movimientos en vez de eventos. Además, recuerda que no queremos que el personaje atraviese las paredes, con lo cual procedemos a detectar las paredes y retroceder si eso ocurre.

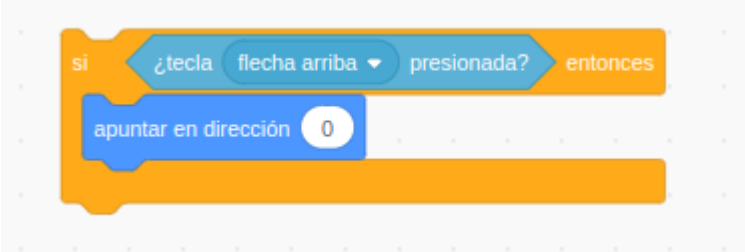
Programaremos la función Inicializar, donde primero lo que haremos será esconder a PacMan y llevarlo a un punto concreto que la pantalla, acuérdate ir a una posición x e y.

Y también programaremos la función Comenzar juego, donde le diremos a Pacman hacia donde moverse con las flechas del teclado.

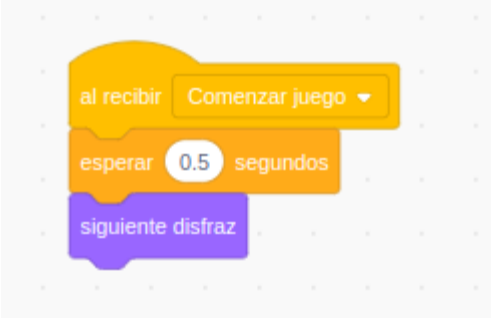


Como puedes ver, te acabo de dar el código completo pero, a cambió, te pido que me describas a continuación lo que ocurre en este bloque.

Pero, si lo compruebas, esto solo hace que PacMan se mueva directamente hacia arriba, así que si queremos cambiar la dirección, tentredmos que crear una nueva función “Comenzar juego” donde le indicaremos que cambie de dirección. Para ello, usaremos bajo la ficha de Comenzar juego, un bucle infinito (es decir, un Por siempre) que anidará cuatro condiciones que varían la dirección de movimiento. Te dejo las fichas para una dirección en concreto, las otras tres puedes duplicar esta y hacer los cambios pertinentes.

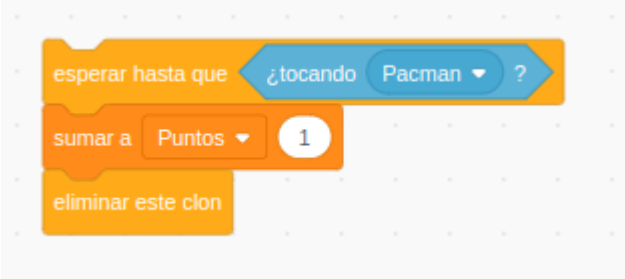


Hasta ahora, nuestro PacMan se mueve automáticamente y cambia de dirección cada vez que pulsamos una flecha pero, ahora haremos que parezca que va comiendo, y lo haremos cambiando de disfraz en una nueva función Comenzar Juego.



Ya que conseguimos mover y redireccionar a nuestro PacMan, ahora tenemos que hacer que se coma la comida. Esto lo programaremos en el Sprite Comida (¡OJO!).

Añadiremos a la función Comenzar clon, una espera a tocar a PacMan, sumar entonces un punto a la variable Puntos y, por último, eliminar el clon.



Con esto, los puntos de comida irán desapareciendo cuando PacMan pase sobre ellos y, además iremos sumando puntos.

¡Venga! Estamos llegando al final, programar los fantasmas.

Algoritmo del fantasma

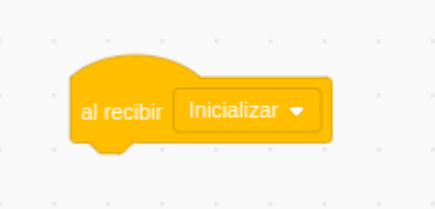
Llegamos a la última parte y, quizás la más compleja y que os cueste algo más comprender, ya que usaremos un algoritmo pseudo-aleatorio.

La funcionalidad de los fantasmas es un algoritmo pseudo-aleatorio, lo que significa que realmente no es aleatorio ya que cada cierto tiempo apuntamos en dirección del objeto principal Pac-Man. El objetivo es que los fantasmas sigan al personaje para que sea más complicado ganar la partida.

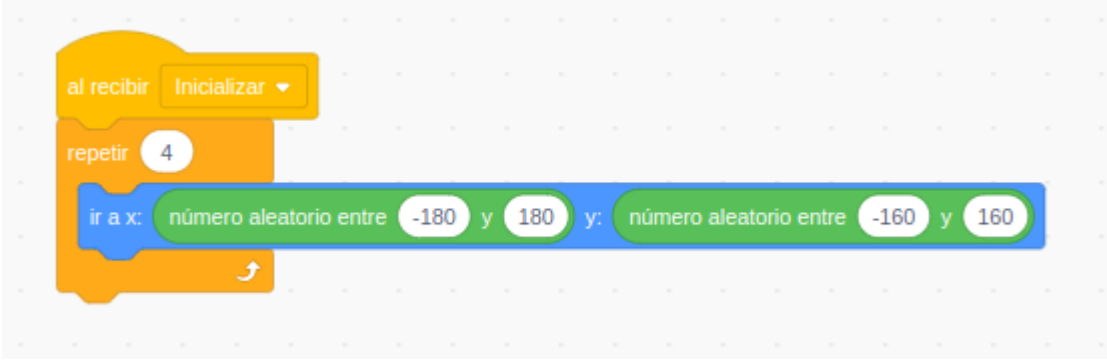
Lo primero será crear el sprite Fantasma, que en el programa base que has usado para comenzar este videojuego ya lo tienes creado, aunque puedes crear el tuyo propio. Lo único a tener en cuenta es que debe tener tres, cuatro o cinco disfraces, según el número de fantasmas que quieras que aparezcan. Yo te lo explicaré con cuatro.

La funcionalidad del Fantasma, y por lo tanto su programación, será que aparezcan cuatro fantasmas en diferentes posiciones aleatorias y, además, que no estén sobre ninguna zona azul de las paredes (aunque se supone que los fantasmas son capaces de atravesarlas). Además, cada uno de esos fantasmas tendrá un disfraz diferente, es decir, un color diferente. Vamos a ello.

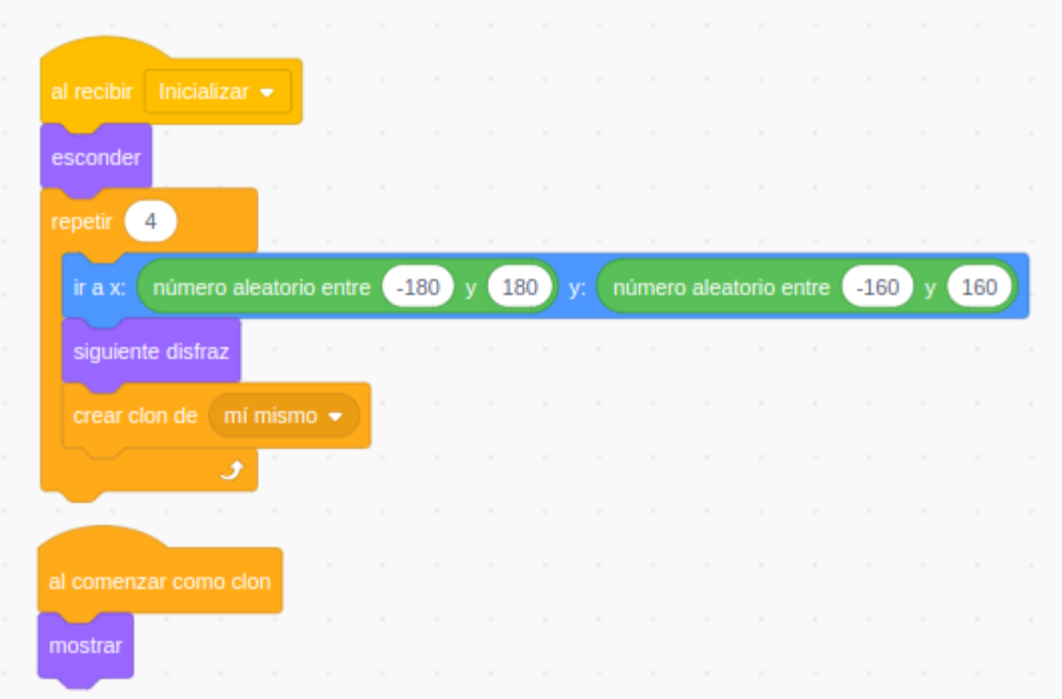
La primera ficha será la de Inicialización.



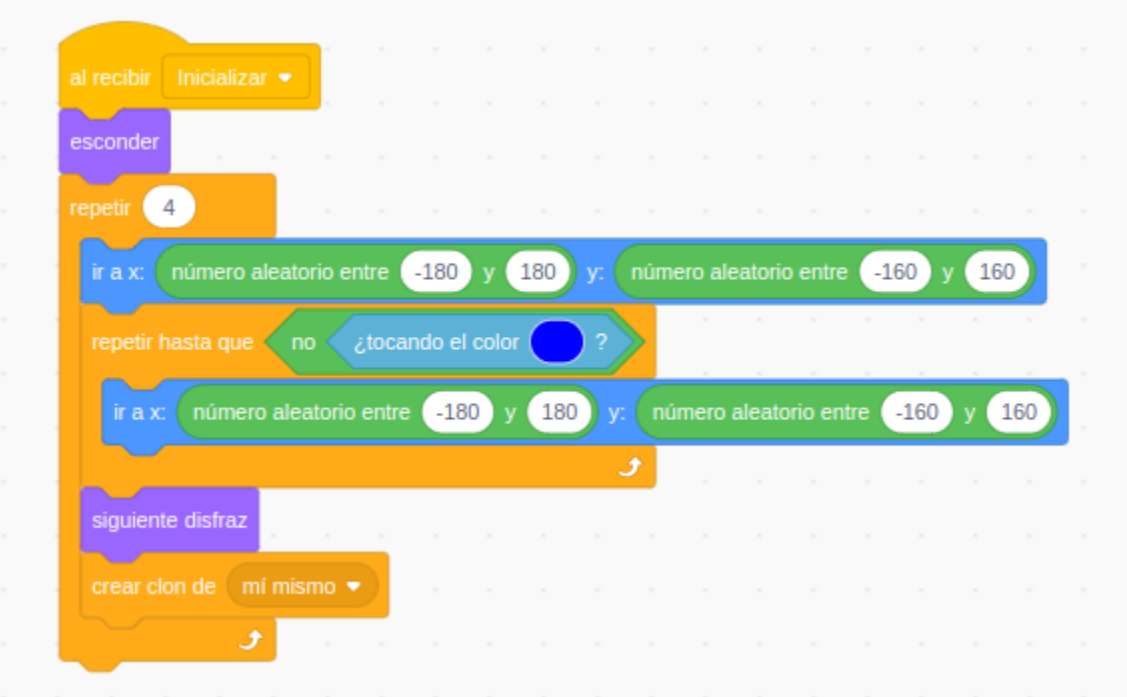
Aquí incluiremos un bucle finito, es decir, que se repita un número concreto de veces, en nuestro caso 4. Éste bucle incluirá la fica de Ir a x e y pero, indicando una posición aleatoria, quedando así:



Ahora tendremos que incluir una llama para crear clones pero, antes, debemos especificar que se esconda para aparecer cuando se llame al clon. Además, que cada uno cambie de disfraz para que sea de diferente color.

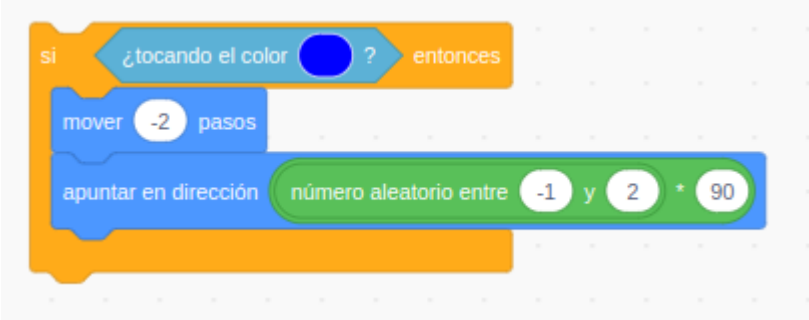


Ahora ya me habrán aparecido los cuatro fantasmas en el juego pero, no podemos permitir que ninguno de ellos aparezca sobre las paredes del laberinto, o lo que es lo mismo, tocando el color azul. Para ello insertamos un código que me repita lo de la posición aleatoria mientras esté tocando la zona azul.

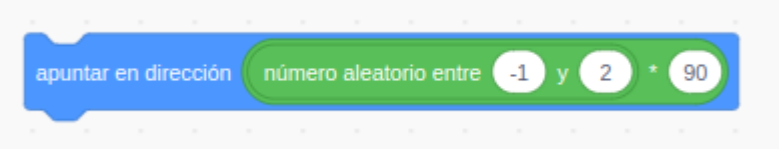


Ahora los fantasmas deben de moverse, ¿no? Pues para ello, seguimos programando después de la ficha Mostrar, usando un bucle infinito (Por siempre), donde le diremos que se mueva y, como en el PacMan, si toca el color azul se mueva en negativo el mismo número de pasos (¿cuántos pasos se moverá? prueba con 2).
Además es bucle debe incorporar un condición, que tocando azul, además cambie la dirección de movimiento de forma aleatoria.

Para hacer esto último y, como queremos que la dirección será aleatoria pero en cuatro direcciones concretas, arriba, abajo, izquierda y derecha, debemos incorporar los siguiente:



¿A que no entiendes el código? sobre todo la siguiente ficha.



Pues lo que hacemos es multiplicar la Dirección, que recuerda que la iniciamos en 90, porun número aleatorio entre -1 y 2.

90 x -1 = -90 ----------> cambiará a dirección a la izquierda
90 x 0 = 0 ----------> cambiará a dirección a arriba
90 x 1 = 90 ----------> cambiará a dirección a la derecha
90 x 2 = 180 ----------> cambiará a dirección a abajo

Bien, ya he creado mis cuatro fantasmas, se mueven y cambian de dirección cuando tocan una pared, ¿que falta? Pues que me pueden matar, y eso ocurrirá cuando cualquiera de ellos toquen a PacMan.

Esta funcionalidad del juego la programaremos en el sprite de PacMan, recuerda que sprite es lo mismo que si dijéramos personaje. Y lo haremos incluyendo una nueva función Comenzar juego que incluirá un bucle Para Siempre que incluirá la condición de que si tocando a Fantasma, entonces, enviar mensaje Game Over.

Ya debes saber lo suficiente como para no tener que ayudarte con imágenes, así que esto último hazlo solo y prueba a ver si funciona, choca con un fantasma y a ves si sale Game Over.

Función extra

Con lo anterior, mi video juego PacMan estará terminado pero, te propongo un reto, que cada cierto tiempo, por ejemplo 3 segundos, los fantasmas cambien de dirección y se orienten hacia PacMan, esto le añadirá un grado más de dificultad, ya que los fantasmas están buscando cazar a nuestro personaje principal.

Te dejo las fichas necesarias, ahora el orden y el sprite donde las debes programar debe ser decisión tuya.

También podemos añadir una dificultad creciente, es decir, que el juego se complique conforme avanza el juego, por ejemplo, que los fantasmas se muevan más rápido conforme vamos comiendo comida, cada 10 puntos de comida, la velocidad del fantasma aumenta en dos puntos más.

Para realizar esto necesitarás conocer una ficha más:

El operador módulo, por definición, lo que hace es entregar el resto de una división entre dos números, entonces, como me entrega el resto de la división entre dos números yo, por ejemplo, podría usarlo para ver si un número es divisible por otro, es decir, cuando un número divide a otro y el resto de la división entera es cero.

Por ejemplo, si yo quiero ver si un número es par, veo si el número módulo 2 da cero, es decir, si el resto de la división por 2 de ese número me da cero, es claramente un número par, en caso contrario es un número impar.

Para nuestro programa, la aplicación que haremos de módulo es, si al dividir por 10 el resto es cero, la velocidad aumentará en uno, entonces, cuando PacMan haya comido 10 puntitos de comida, el valor de la variable Puntos será 10, que si dividimos por 10, su resto será cero, aumentará entonces la velocidad a 3, ya así sucesivamente, con 20 puntos, la división por 10 dará como resto cero, volverá aumentar,....