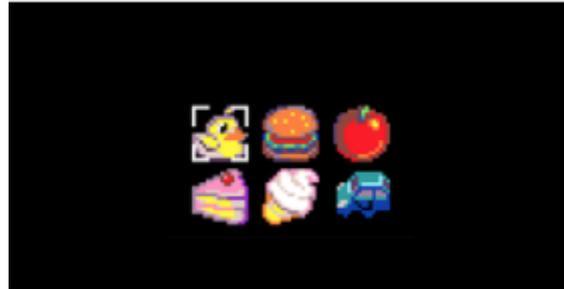


Memory



Cuando empieza

```
on start
  set index to 0
  set value to 0
  show long text "Memoriza la secuencia y reproducela" bottom
  set spritelist to array of sprites of kind glyph
  set score to 0
  set life to 3
  call InitImages
  call InitSounds2
  call InitUI
  call InitSequence
  call PlaySequence
```

La variable index le asignamos el valor 0.

La variable value le asignamos el valor 0.

Muestra el texto largo en la parte inferior de la pantalla,

Asignar a variable score (puntos) el valor inicial a 0.

Asignar a la variable life (vidas) el valor inicial de 0.

Llamar a funciones.

InitImages: Almacena en la array `imageList` las 6 imágenes que mostrará el juego.

InitSounds2: almacena una serie de 6 números que servirá para reproducir sonido.

InitUI: Muestra los 6 elementos del juego en la parte inferior con su respectivo cursor para poderlas seleccionar.

Función InitImages.



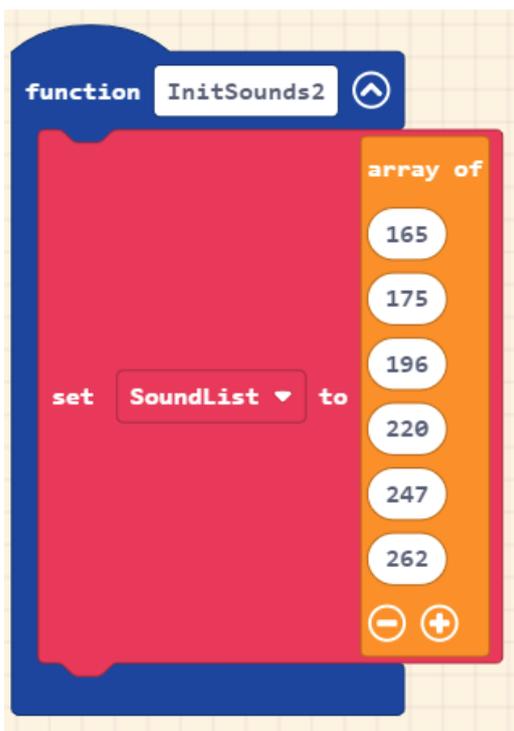
InitSequence: realiza un bucle que se repite tres veces y llama tres veces a la función AddToSequence, esta a su vez almacena al final de la array CodeSequence un valor aleatorio de 0 hasta 5.

PlaySequence: Muestra la nueva serie de objetos a memorizar durante 600 ms.

Definimos una array llamada imageList que almacenará 6 imágenes.

Todas están disponibles en la galería.

Función InitSound2



Definimos una array llamada SoundList que almacenará 6 valores numéricos que utilizaremos a posteriores para reproducir sonidos.

Función InitUI

Al Sprite mySprite le asignamos la imagen del array imagList el elemento 0, es de tipo UI.
Dicho Sprite lo posicionamos en las coordenadas x: 60, y: 90.

```
function InitUI
  set mySprite to sprite imagList get value at 0 of kind UI
  set mySprite position to x 60 y 90
  set mySprite to sprite imagList get value at 1 of kind UI
  set mySprite position to x 80 y 90
  set mySprite to sprite imagList get value at 2 of kind UI
  set mySprite position to x 100 y 90
  set mySprite to sprite imagList get value at 3 of kind UI
  set mySprite position to x 60 y 108
  set mySprite to sprite imagList get value at 4 of kind UI
  set mySprite position to x 80 y 108
  set mySprite to sprite imagList get value at 5 of kind UI
  set mySprite position to x 100 y 108
  set CursorSprite to sprite [ ] of kind UI
  set CursorSprite position to x 60 y 90
  set curX to 0
  set curY to 0
```

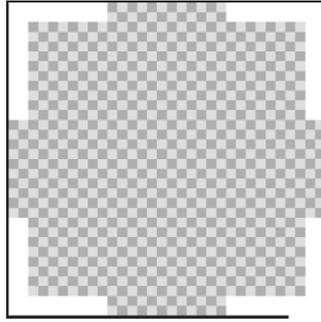
Lo repetimos con el resto de objetos.

Creamos Sprite de tipo UI con el nombre CursorSprite.

Lo posicionamos en las coordenadas x: 60, y: 90.

A la variable curX le asignamos el valor 0.

A la variable curY le asignamos el valor 0.



Con una dimensiones de 16 x 16

La función InitSequence

```
function InitSequence
  for index3 from 0 to 2
  do
    call AddToSequence
```

Realizamos un bucle que se repite 3 veces, por este motivo llama tres veces a la función AddToSequence.

Esta función añade al final un valor aleatorio desde 0 hasta 5 a la array CodeSequence.

La función PlaySequence

```
function PlaySequence
  for index2 from 0 to length of array CodeSequence - 1
  do
    set CurrentIndex to index2
    call AddToSpriteList
    pause 200 ms
  pause 600 ms
  for element value2 of spriteList
  do
    destroy value2
  set CurrentIndex to 0
  set PlayerTurn to true
```

Un ciclo for que se repite el número de elementos del array CodeSequence menos 1.

A la variable CurrentIndex le pasamos el valor del índice actual.

Llamamos a la función AddToSpriteList más pausa 200 ms.

La función AddToSpriteList muestra de nuevo los elementos a memorizar durante 600 ms.

Pausa de 600 milisegundos.

Destruye todos los spriteList, que nos está mostrando.

La variable CurrentIndex pasa a valer 0.

La variable PlayerTurn pasa a valer true.

AddToSpriteList

The code block for the `AddToSpriteList` function consists of the following steps:

- Set `x` to `remainder of CurrentIndex / 9 * 17`.
- Set `y` to `floor of CurrentIndex / 9 * 17`.
- Set `mySprite` to `sprite imagenList get value at CodeSequence get value at CurrentIndex of kind glyph`.
- `spriteList set value at CurrentIndex to mySprite`.
- Set `mySprite left` to `x + 5`.
- Set `mySprite top` to `y + 20`.
- Play tone at `SoundList get value at CodeSequence get value at CurrentIndex` for `1/2 beat`.

Annotations explaining the code:

- A la variable `x` se le asigna el valor de `Remaider(CurrentIndex / 9) x 17`.
- A la variable `y` se le asigna el valor de `floor (currentmIndex / 9) x 17`.
- Son las coordenadas para los objetos que va mostrando.
- Crea un Sprite de nombre `MySprite` con la imagen según `CodeSequace` (valor de 0 hasta 5) según `CurrentIndex` (Índice actual) de tipo `glyph`.
- `SpriteList` será igual `CurrentIndex` (Índice Actual) de `mySprite`.
- La posición del `mySprite` por la izquierda será el valor de `x + 5`. La posición del `mySprite` por la derecha será el valor de `y + 20`.
- Se reproduce un sonido, según valor aleatorio desde 0 hasta 5 del índice actual por medio latido.

Remaider: Este es un operador adicional para la división. Puede averiguar cuánto queda si un número no se divide en el otro número de manera uniforme.

Sabemos que $4/2$ es igual a 2 por lo que se divide en 4 uniformemente. Pero $5/2$ es igual a 2 pero con un resto de 1. Entonces la operación de `resto%2` es igual a 1, del número que queda de una operación de división.

Floor: Para hacer que un número cambie al siguiente número entero más bajo (entero), obtenga el valor mínimo del número. El valor mínimo de 8,76 es 8, ya que es el siguiente número entero más bajo. Para el número negativo de 6,17, su piso es -7 ya que es el siguiente número entero más bajo.

Función AddSequence

The code block for the `AddToSequence` function consists of the following step:

- `CodeSequence add value pick random 0 to 5 to end`

La variable `CodeSequence` se le asigna un valor aleatorio desde 0 hasta 5

Función UpdateCurPos

```
function UpdateCurPos  
  set CursorSprite x to 60 + 20 * curX  
  set CursorSprite y to 90 + 18 * curY
```

A la variable CursorSprite de x se le asigna el valor de $(60+20x(\text{valor de curX}))$.

A la variable CursorSprite de y se le asigna el valor de $(90+18x(\text{valor de curY}))$

Función NextLevel

```
function NextLevel  
  set PlayerTurn to false  
  pause 500 ms  
  for element value3 of spriteList  
  do  
    destroy value3  
  call AddToSequence  
  call PlaySequence
```

La variable PlayerTurn pasa a valer false.

Una pausa de medio segundo.

Un bucle que se repetirá según el número de elementos de spriteList.

Los destruye a todos los elementos.

Llamamos a la función AddSequence.

Llamamos a la función PlaySequence.

Cuando se crea un Sprite de tipo UI

```
on created sprite of kind UI  
  set sprite ghost ON
```

Sprite en modo fantasma.

Cuando se presiona el botón A

The image shows a Scratch code block for the event "on A button pressed". The code is as follows:

```
on A button pressed
  if PlayerTurn = true then
    set CursorAt to curX + curY x 3
    if CursorAt = CodeSequence get value at CurrentIndex then
      call AddToSpriteList
      change score by 1
      change CurrentIndex by 1
      if CurrentIndex = length of array CodeSequence then
        call NextLevel
      else
        change life by -1
        play sound power down
```

Annotations with arrows pointing to specific parts of the code:

- "Si es turno del jugador." points to the first `if` block.
- "La variable CursorAt se le asigna el siguiente valor: $curX + curY \times 3$." points to the `set CursorAt` block.
- "Si CursorAt es igual al valor de la array CodeSequence en el elemento CurrentIndex." points to the second `if` block.
- "Llamamos a la función AddToSpriteList." points to the `call AddToSpriteList` block.
- "Score y CurrentIndex incrementa en 1" points to the `change score by 1` and `change CurrentIndex by 1` blocks.
- "Llamar a la función NextLevel." points to the `call NextLevel` block.
- "Si CurrentIndex es igual al número de elementos de CodeSequence" points to the third `if` block.
- "Si no" points to the `else` block.
- "A la variable life se le resta 1." points to the `change life by -1` block.
- "Se reproduce un sonido 'power down'" points to the `play sound power down` block.

Cuando se presiona el boto de abajo.

The image shows a Scratch code block for the event "on down button pressed". The code is as follows:

```
on down button pressed
  set curY to min of 1 and curX + 1
  call UpdateCurPos
```

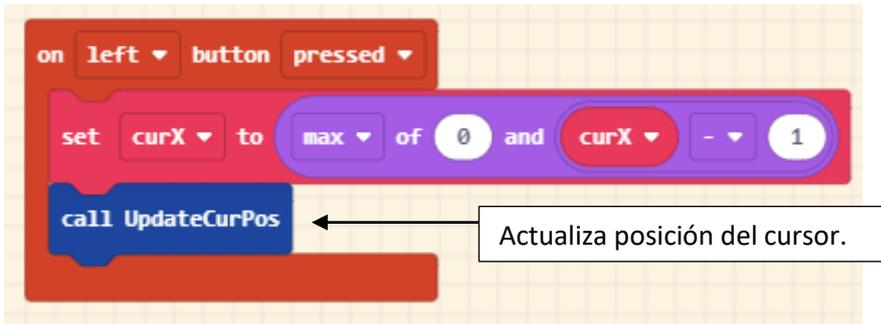
Annotation with an arrow pointing to the `call UpdateCurPos` block:

- "Actualiza posición del cursor." points to the `call UpdateCurPos` block.

La variable curY obtiene el valor mínimo de 1 y $curX + 1$.

Llama a la función UpdateCurPos.

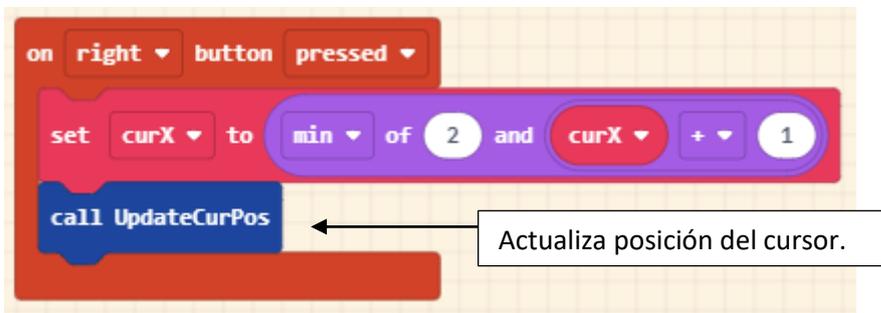
Cuando se presiona el botón Izquierda



La variable curX obtiene el máximo de 0 y $curX - 1$.

Llama a la función UpdateCurPos.

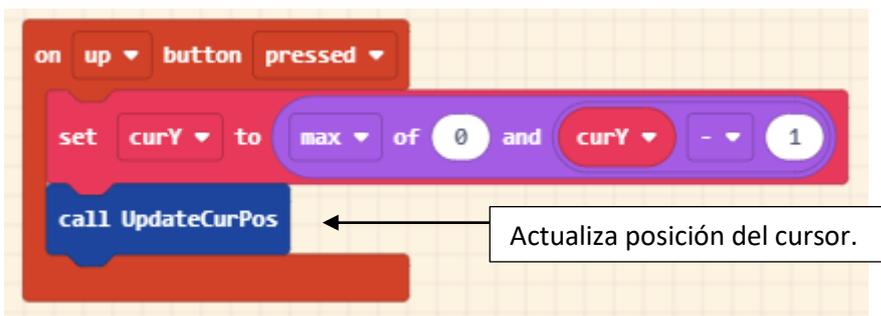
Cuando se presiona el botón derecha.



La variable curX obtiene el mínimo de 2 y $curX + 1$.

Llama a la función UpdateCurPos.

Cuando se presiona el botón arriba



La variable curY obtiene el máximo de 0 y $curY - 1$.

Llama a la función UpdateCurPos.

Mínimo y máximo de dos valores: Puede obtener el menor o el mayor de dos números con las funciones `min()` y `max()`.

El mínimo de 2 y 9: `Math.min(2,9)` es igual a 2.

El máximo de 3 y 9: `Math.max(3,9)` es igual a 9.