

# Práctica guiada 4. Piedra, papel o tijera


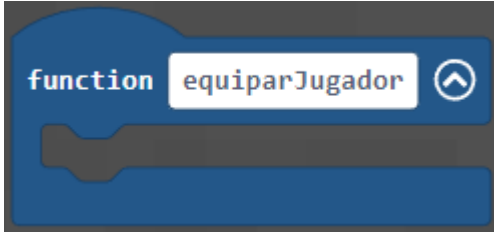
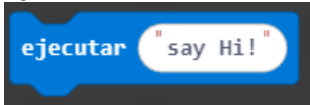
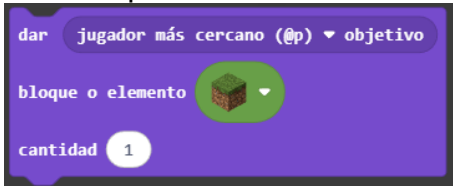
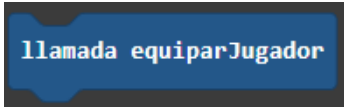

En esta práctica aprenderemos a diseñar y programar un minijuego de tipo “Piedra, papel o tijera” dentro de Minecraft Education. A lo largo de las siguientes actividades, construiremos la lógica del juego utilizando variables, condicionales y aleatorización. Programaremos la respuesta del agente mediante bloques de chat, generaremos jugadas aleatorias y compararemos resultados para determinar el ganador de cada ronda. Además, exploraremos el uso de estructuras condicionales anidadas y cómo estructurar un sistema básico de decisión dentro del entorno de programación por bloques.


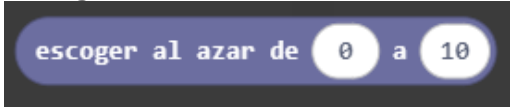
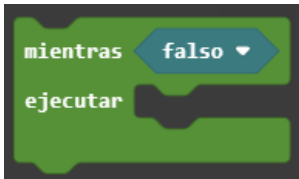


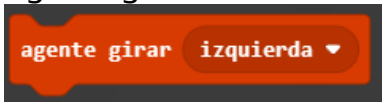
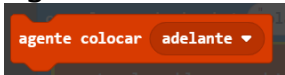

## Mecánicas de Minecraft

Mecánicas	Para ordenador
Abrir el editor de código	C
Abrir chat	T
Abrir el menú de pausa; cerrar los menús de juego	ESC
Minar un bloque	Clic Izquierdo
Interactuar con botones y con personajes	Clic derecho
Saltar	Barra espaciadora
Movimiento del jugador (Adelante/Atrás/Izquierda/Derecha)	W/S/A/D
Dirección del jugador	Movimiento del ratón

## Bloques para esta práctica

<b>Variables</b> 	Son contenedores de información.
<b>Fijar</b> 	Establece el valor para una variable.

<p>Valor del bloque</p> 	Devuelve un tipo concreto de bloque de Minecraft.
<p>Función</p> 	Bloque de código que realiza una tarea en específico.
<p>Ejecutar</p> 	Ejecuta comandos de chat a través de código.
<p>Dar bloque o elemento</p> 	Coloca el bloque o elemento seleccionado, la cantidad de veces seleccionada al inventario del jugador escogido.
<p>Con el comando de chat</p> 	Evento que se activa al escribir una palabra en concreto en el chat.
<p>Llamada a una función</p> 	De este modo se puede ejecutar el código de una función.
<p>Agente coloca bloque u objeto</p> 	Coloca el número especificado de bloques de minecraft/objetos en la ranura especificada del inventario del agente.
<p>Agente desplazarse</p> 	Mueve al agente en la dirección especificada el

	número de veces indicado.
<p>Establecer ranura activa</p> 	Establece que ranura del inventario va a utilizar el agente.
<p>Escoge al azar</p> 	Selecciona al azar un valor entre el intervalo introducido.
<p>Mientras</p> 	Mientras que no ocurra la condición, el bucle se seguirá ejecutando, una vez que ocurra, parará.
<p>No</p> 	Invierte una condición lógica.
<p>Agente detectar</p> 	Comprueba si hay un bloque en la posición seleccionada respecto al agente.
<p>Agente girar</p> 	El agente modifica su dirección dependiendo de lo que se le pida.
<p>Agente colocar</p> 	Coloca un objeto en el mundo desde la ranura seleccionada anteriormente del inventario del agente.
<p>Agente destruir</p> 	Manda al agente la orden de destruir el bloque

	que se encuentre en la dirección indicada.
Pausa 	Para la actividad del Código durante los (ms) escogidos.
Agente inspeccionar 	Inspecciona un bloque en la dirección indicada, devuelve el tipo de bloque.
Si / si no 	Permite ejecutar acciones diferentes según si se cumple o no una condición.
Condicional 	Devuelve verdadero si los dos valores de la condición son iguales.
Mostrar 	Muestra un texto en pantalla al jugador seleccionado, con la opción de poner un título y un subtítulo.

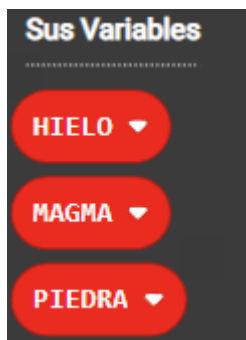
## Parte 1. Adaptando el juego

Comenzamos creando un nuevo proyecto para el juego, recordando siempre poner un nombre característico.

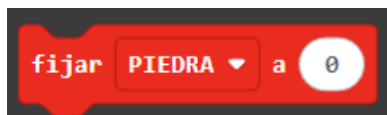
Una vez tenemos creado el proyecto, deberemos tener en cuenta que, para poder jugar piedra, papel o tijera, contra el agente, vamos a necesitar tener esos elementos, pero no hay bloques que representen el papel o las tijeras, por tanto, los bloques que usaremos serán: bloque de magma, hielo y piedra. De forma que el magma derrite la piedra, la piedra rompe el hielo y el hielo enfría el magma.



Tal y como hicimos en la práctica anterior, vamos a utilizar constantes para nuestros tres tipos de bloque. ¿Recordáis cómo hacerlo? Así deben quedar:



Además de crearlas, también tendremos que establecer qué es cada una, haz memoria de la práctica anterior. Deberemos gastar el bloque “fijar” y el bloque de “valor del bloque”, debéis recordar en qué parte del código se colocaba.



Como no estaremos utilizando piedra, papel y tijeras, como estamos acostumbrados, hay una característica de la programación que puede resultar muy útil cuando estamos programando, y es poner comentarios. Esta herramienta puede ser muy útil para recordar que hacía tu código o para hacérselo saber a otras personas que vayan a trabajar en él. Para poder añadir comentarios deberemos hacer clic derecho sobre nuestro bloque y hacer clic en añadir comentario. Como se muestra a continuación:



Una vez le demos podremos poner lo que queramos y quedará de la siguiente manera:



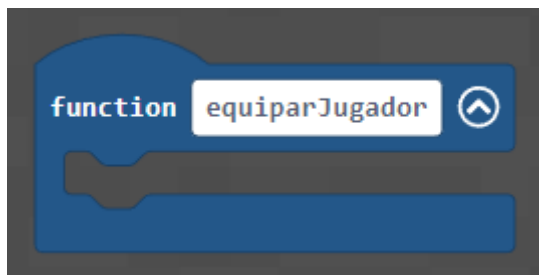
Si no pudieseis ver los comentarios, podéis acceder a ellos desde el símbolo del bocado que se genera a la izquierda del bloque.

Ahora ya tenemos adaptados los elementos que componen el juego de “piedra, papel o tijera”, podemos comenzar a programarlo.

## Parte 2. Equipando a los jugadores

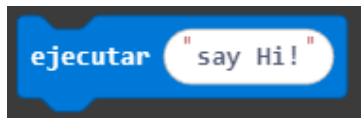
Para poder organizar las diferentes partes del código de una forma más limpia, vamos a comenzar a gastar funciones.

Para crear funciones debemos ir al apartado de FUNCIONES, que se encuentra en AVANZADO, y podremos observar que hay que generarlas como hacíamos con las variables, en este caso, aunque los nombres siempre es necesario que sean relevantes y distintivos, el formato es diferente puesto que se escribe en minúsculas, poniendo en mayúsculas la primera letra de cada palabra después de la primera, es decir, por ejemplo como nombre para nuestra primera función, que dará los bloques necesarios al jugador, podemos llamarla “equiparJugador”. Una vez creada, nos aparecerá lo siguiente en la zona de código.



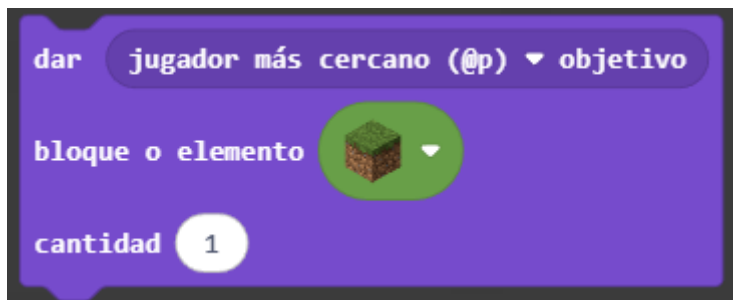
Como se puede observar, un bloque de función no se puede enganchar a nada, pero sí puede contener código que determinará el funcionamiento de la función.

Para esta función en concreto, deberemos dar al jugador, bloques de magma, hielo y piedra, pero primero deberemos asegurarnos de que no tiene ningún otro elemento en el inventario, así que, como en la práctica 1, vamos a vaciar el inventario del jugador con “ejecutar”.

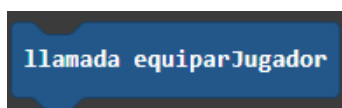
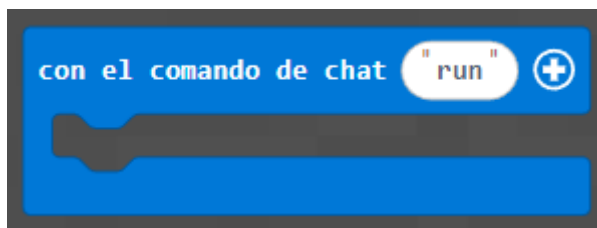


Si no recuerdas el comando adecuado para vaciar el inventario, revisa las prácticas anteriores.

Una vez vaciado el inventario, vamos a dar al jugador los diferentes bloques necesarios, usando el bloque “dar” y cambiando el valor del bloque por nuestros valores de constantes.

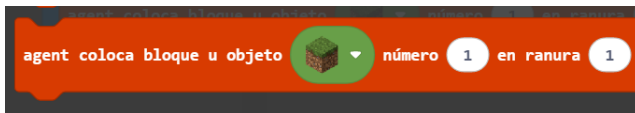


Una vez terminada nuestra función, tenemos que decirle a nuestro programa cuando debe activarla, para ello vamos a utilizar el bloque “con el comando de chat...”, le vamos a cambiar el nombre a “jugar” y le vamos a añadir el bloque que llamará a nuestra función, es decir, que la activará, este último bloque está dentro del apartado FUNCIONES.



Y ya tenemos nuestra primera función completa, prueba a lanzar el código y poner en el chat “jugar” a ver si el código da al jugador los bloques necesarios para jugar.

Tal y como hemos hecho con el jugador, ahora vamos a equipar al agente, para ello vamos a crear una nueva función y la vamos a llamar “equiparAgente”. Y, a continuación, utilizando “agente coloca bloque...” vamos a darle al agente (utilizando las variables constantes), la piedra en la ranura 1, el magma en la ranura 2 y el hielo en la ranura 3.



Para poder probar si funciona, deberemos llamar a la función desde el bloque “con el comando de chat”, tal y como hemos hecho con la función anterior. De forma que, al poner “jugar” en el chat, el programa equipe al jugador y al agente con los bloques necesarios para jugar. Para comprobar el inventario del agente, haz clic derecho sobre él.

### Parte 3. La selección del agente

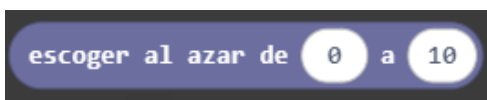
Para poder jugar a piedra, papel o tijera, necesitamos que nuestro agente seleccione uno de los elementos que le hemos dado al azar. La forma en que va a funcionar el juego es la siguiente, el jugador colocará un bloque justo debajo del agente, y cuando lo haga, el agente responderá colocando un bloque aleatorio, de los tres que tiene encima suyo. Después, analizará que bloque gana la partida, el de arriba o el de abajo, y según quien gane, se lo comunicará al jugador.

Pero para comenzar, haremos una nueva función que hará que el agente escoja un elemento al azar, la llamaremos “agenteEscoge”.

Como hemos dicho, el agente va a colocar un bloque encima suyo, y el jugador lo colocará justo debajo, por lo que necesitamos que el agente esté posicionado un bloque más arriba del nivel del suelo, podemos hacer esto con el bloque “agente desplazarse”, dando la instrucción hacia arriba.

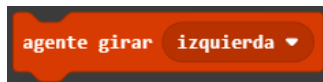


Para que el agente seleccione uno de los bloques que tiene de forma aleatoria, debemos combinar los siguientes dos bloques, el primero, “agente establecer ranura activa”, que determina que ranura del inventario es la que va a utilizar el agente, y el bloque “escoger al azar”, que nos permitirá que el agente escoja un bloque de forma aleatoria que esté entre sus ranuras 1 y 3.



Ahora que ya tenemos la forma de que el agente seleccione al azar un bloque, no podemos pedirle que lo coloque hasta que el jugador haya colocado el suyo, porque entonces el jugador vería la selección del agente y haría trampas para ganar. Por tanto, debemos hacer que el agente coloque su bloque después de que el jugador lo haya colocado. ¿Cómo haremos esto? Pues con un bucle, mientras que el agente no detecte ningún bloque debajo suyo, el agente esperará, pero para que no parezca que el código no funciona, haremos que el agente gire mientras que espera. Para hacer todo esto, necesitaremos los siguientes bloques:





Una vez el jugador haya puesto su bloque seleccionado debajo del agente, el bucle parará y el código continuará a la siguiente línea, por lo que ya podremos decirle que coloque el bloque seleccionado encima suyo.



Solo nos quedaría llamar a la función, como hechos hecho con las anteriores y comprobar si funciona colocando un bloque debajo del agente.

## Parte 4. Reseteando el juego

Con las pruebas, te habrás dado cuenta de que cada vez que quieres comenzar la partida deber ir a resetear el mapa para que el agente se vuelva a colocar en la posición adecuada, para evitar tener que hacer esto, y antes de ponernos con la parte de dictaminar el ganador, vamos a añadir una nueva función que se llamara “resetearJuego”, para facilitarnos las pruebas que estemos haciendo.

Si tu espacio de trabajo está quedándose muy lleno por todas las funciones utilizada, puedes hacer clic en la flecha blanca que hay junto al nombre de cada una, de esta forma, se compactarán y podrás trabajar mejor. Deberían quedarte así:



Con el espacio de trabajo limpio, ya podemos dedicarnos a resetear el juego. Lo primero que vamos a decirle a la función es que el agente destruya los bloques que tiene justo encima y justo debajo. Y después de este, le diremos que se mueva un bloque hacia

abajo, igual que hacíamos en la función anterior para decirle que subiese, pero hacia abajo.



Ahora, vamos a llamar a la función como hemos hecho con todas las anteriores, pero, nos surge un problema, y es que el código va muy rápido y no daría tiempo al jugador para visualizar los resultados antes de que el agente los destruya, por esta razón, vamos a colocar al comienzo de nuestra función, y antes de que el agente destruya nada, un bloque de pausa, le podemos dar los segundos que creamos necesarios, pero recordad que el bloque trabaja con milisegundos, por lo que 1 segundo serán 1000 milisegundos.



## Parte 5. Determinar el ganador

Comenzaremos con una nueva función, a la que llamaremos “determinaGanador”. Para determinar el ganador deberemos saber que bloque ha colocado el jugador y cual ha colocado el agente, para almacenar esta información utilizaremos dos nuevas variables, pero estas no serán variables constantes, como hemos estado trabajando, porque su valor no está determinado antes de lanzar el código, si no, que se determina a lo largo del código, por esto, el estilo de nombre es diferente, y es igual que el que hemos estado usando para las funciones. Por esto, nuestras variables se llamarán de la siguiente forma “bloqueJugador” y “bloqueAgente”.

Además, a diferencia de las anteriores variables, estas no se fijan, o se inician, en el bloque “al iniciar”, si no, que se crean al inicio de nuestra función de “determinarGanador”, y el valor que recibirán las variables dependerá de lo que haya colocado encima y debajo del agente, utilizando el bloque “agente inspeccionar”.



Ahora, que ya tenemos guardada en una variable la información de cada bloque, podemos pasar a las reglas que nos ayudarán a determinar qué bloque gana a qué. Teniendo en cuenta que cada jugador tiene tres opciones entre las que elegir, deberemos programar de tal forma que se quede lo siguiente: Si el jugador tiene ... y el agente tiene ..., el resultado es ..., algo como la siguiente tabla de relaciones:

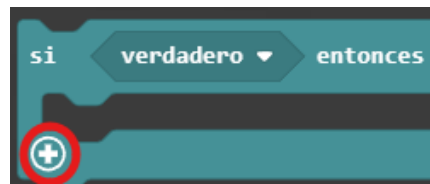
Jugador \ Agente			
	PIEDRA	MAGMA	HIELO
PIEDRA	Empate	Agente gana	Jugador gana
MAGMA	Jugador gana	Empate	Agente gana
HIELO	Agente gana	Jugador gana	Empate

Pero no os asustéis, no habrá que hacer un “si” por cada una de las 9 posibilidades, de hecho, si nos fijamos, hay 3 posibilidades que son exactamente iguales, pase lo que pase, puesto que, si los dos jugadores sacan el elemento igual al otro, quedarán empate, independientemente del bloque que sea. Vamos a comenzar por esto.

Tendremos que hacer un “si” el bloque del agente es igual al bloque del jugador, entonces, quedan empate. Para ello necesitaremos los bloques que se muestran a continuación y vosotros ya los organizáis como creáis conveniente, en base a lo que ya hemos dado.



Ahora que ya tenemos los empates, vamos a trabajar con las otras posibilidades. Para ello, necesitamos añadir nuevas posibilidades a nuestro condicional “si”, lo haremos haciendo clic al símbolo de más que tiene el bloque.



Como vamos a tener múltiples opciones, vamos a darle tres veces al más, creándonos nuevas opciones de condición.



Estas nuevas opciones nos proporcionarán la posibilidad de Si el jugador ha seleccionado un bloque igual a piedra, si ha seleccionado un bloque igual a magma o si no ha seleccionado ninguno de esos dos bloques y, por tanto, ha seleccionado el hielo. Para

realizar las dos comparaciones que se comentan, recuerda utilizar las variables ya creadas.

Ahora, pongámonos en la parte del condicional de que el jugador ha seleccionado piedra, si ese es el caso, quiere decir que el agente NO ha seleccionado piedra, porque se hubiese activado el empate, por tanto, ha seleccionado o magma o hielo. Así que dentro del “si jugador selecciona piedra”, deberemos tener dos posibles salidas con un nuevo “si”, si el agente escoge magma entonces el agente ganará y si el agente escoge hielo, entonces ganará el jugador. Rellena con el bloque condicional de “si ... sino” correspondiente y recuerda notificar al jugador del resultado con un bloque de “mostrar”, como hemos hecho anteriormente. Haz este mismo pensamiento lógico para el resto de las posibilidades del jugador, es prácticamente copiar y pegar el código.

Una vez finalizado, ya puedes lanzar el código y probar el juego.

## Entrega

Una vez hayas finalizado el juego, asegúrate de:

1. Guardar el mundo correctamente.
2. Guarda el proyecto y realiza una captura de pantalla del código y del circuito



3. Presenta de forma oral el proyecto o haz un documento con las capturas y la explicación del código realizado.
4. Sube el archivo del proyecto y el documento, si es el caso, a Moodle.

**ANTES DE LA PRÓXIMA PRÁCTICA.**