Raspbian

APRENDE
Y DIVIÉRTETE

GOBIERNO DE LA CIUDAD DE MÉXICO

SECTEI

CIUDAD **INNOVADORA** Y DE **DERECHOS**

Módulo 3

Semana	Tiempo sugerido	Temas/ Subtemas	Aprendizaje esperados	Eje -Ámbitos-Ambientes Sociales de Aprendizaje
10	70-90 minutos	Platicando con un Viajero Espacial	Reconoce tipos de sentencias aritméticas que se utilizan en los lenguajes de programación para evaluar una operación lógica.	Número, Álgebra y Variación.
11	70-90 minutos	Juego Mental	Estimula la habilidad numérica. Programa un juego basado en las tablas de multiplicar.	Número, Álgebra y Variación. Análisis de Datos.
12	90-120 minutos	Guerra de Galaxias	Conoce herramientas de Scratch para simular los movimientos de una nave espacial de derecha a izquierda, de esa forma se familiariza gráficamente con los eventos que se pueden realizar en un lenguaje de programación, desarrolla competencias referentes al cálculo aproximado de área de figuras.	Número, Álgebra y Variación. Análisis de Datos.
13	90-120 minutos	Carrera de Coches	Programa un juego en el que interviene forma, medida y condicionamientos lógicos.	Número, Álgebra y Variación. Análisis de Datos.

10. Platicando con un viajero espacial

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Crea y utiliza bloques de Scratch personalizados. Usa una variedad de operadores lógicos y de comparación. Usa variables y estructura de datos simples.	Por medio de un IDE: Scratch.	Crea tu viajero espacial. Hace hablar a un viajero espacial. Toma decisiones a partir de los datos que se reciban. Cambia su ubicación.	Desarrollar una práctica en la que se logre interactuar con una animación.

10. Platicando con un Viajero Espacial

Introducción

¡Vas a aprender a programar un Viajero Espacial! Harás que exista un diálogo con tu viajero y según lo que digas en el diálogo, lo harás responder, cambiar su apariencia y moverse en el escenario.

Comencemos.

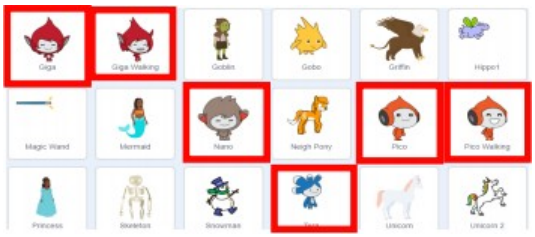
Crea tu Viajero Espacial

Antes de empezar a crear tu viajero espacial, necesitas decidir sus características:

¿Cómo se llama?, ¿Dónde vive?, ¿Es feliz?, ¿Es serio? ¿Es gracioso?, ¿Tímido?, ¿Amigable?

Crea un nuevo proyecto de Scratch.

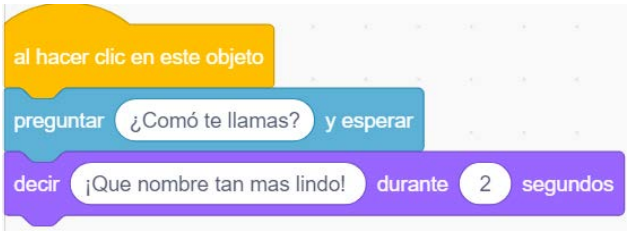
Elige uno de los personajes de este menú de opciones ubicada en objetos que Scratch te ofrece y añade el que más te guste a tu proyecto.

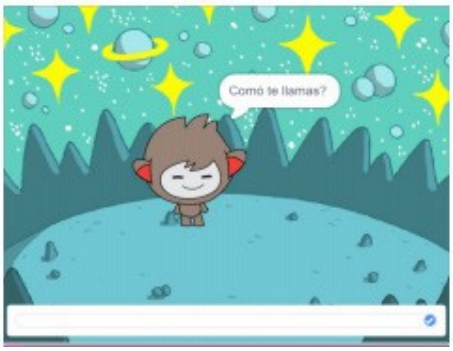


Escoge un escenario que le vaya bien a la personalidad de tu viajero. Aquí tienes un ejemplo, aunque tu escenario puede ser diferente.

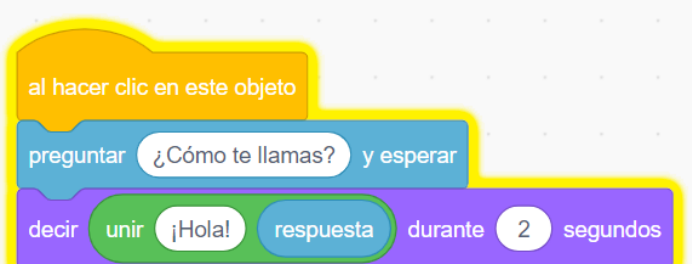
Haz hablar a tu Viajero Espacial

Ahora que ya tienes un viajero espacial con personalidad. Haz clic en el personaje de tu viajero espacial y añade este código.





Haz clic en tu viajero. Después de que te haya preguntado tu nombre, escríbelo en el recuadro que aparece en la parte inferior del escenario. Tu viajero simplemente responderá: ¡Qué nombre más lindo! Muy simple aún. Cambiemos ahora su respuesta:



Por ejemplo, para crear este último bloque, primero tendrás que seleccionar un bloque “unir” (de la sección de **operadores**) y arrastrarlo sobre el bloque “decir**”**.

Puedes cambiar el primer texto por “Hola”, y arrastrar el bloque azul “respuesta” (de la sección de **sensores**) sobre el texto.

Prueba este nuevo programa. ¿Funciona cómo esperabas?

Puede que quieras guardar el nombre del usuario en una variable, para poder usarlo de nuevo en el futuro. Crea una nueva variable que se llame nombre.

La información que has escrito ya está almacenada en una variable especial llamada respuesta. Ve a la sección de bloques **sensores** y haz clic en el bloque de respuesta para que aparezca una marca de verificación.

El valor actual en respuesta debería aparecer en la parte superior izquierda del escenario.

Una vez que hayas creado tu nueva variable, asegúrate de que el código de tu viajero espacial se vea como éste:

Si pruebas el programa una vez más, verás que la respuesta se guarda en la variable “nombre” y aparece en la parte superior izquierda del escenario. La variable nombre debería contener el mismo valor que la variable respuesta.

Toma decisiones

Puedes programar tu viajero espacial para que decida qué hacer en función de las respuestas del usuario. Veamos cómo hacer esto:

Vamos a hacer que tu viajero espacial haga al usuario una pregunta con respuesta: sí o no. Por ejemplo, haz que pregunte “¿Estás bien?”

Para probar este programa adecuadamente, tendrás que probarlo dos veces, una escribiendo “no” como respuesta y otra escribiendo “sí”. Sólo deberías obtener una respuesta de tu viajero si tu respuesta es “sí”.

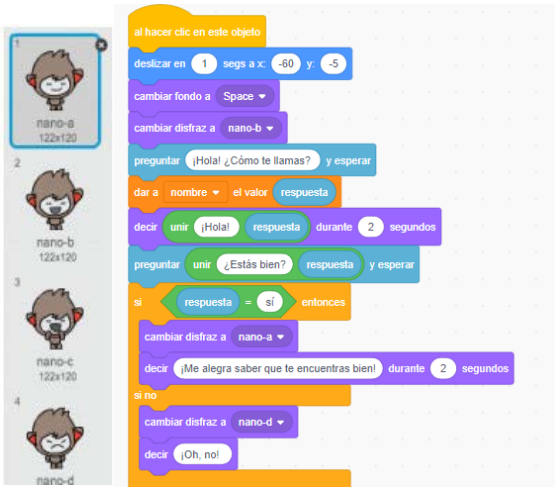
Como habrás notado, el problema con tu viajero es que no te responde nada si el usuario contesta “no”. Puedes solucionar esto cambiando el bloque “**si**” por un bloque “**si/si no**” de modo tal que tu nuevo código se vea como éste:

Si pruebas tu código ahora, verás que obtienes respuesta si contestas: “sí” o “no”. Tu viajero debería responder con ¡Me alegra saber que estás bien! cuando contestas “sí”, pero responderá ¡Oh, no! si escribes cualquier otra cosa (en tu bloque, “**si no**” significa “de otra manera”).

Puedes poner cualquier código dentro de un bloque “**si/si no**”; no solamente código para hacer que tu viajero espacial hable.

Por ejemplo, puedes cambiar el disfraz de tu viajero para que coincida con su respuesta. Si miras los disfraces de tu viajero, verás que hay más de uno.

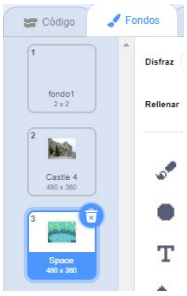
Puedes usar estos disfraces como parte de la respuesta de tu viajero espacial, añadiendo este código:



Cambia su ubicación

También puedes programar a tu viajero para que cambie de ubicación.

Para esto, añade otro fondo a tu escenario, por ejemplo: el fondo de un castillo.

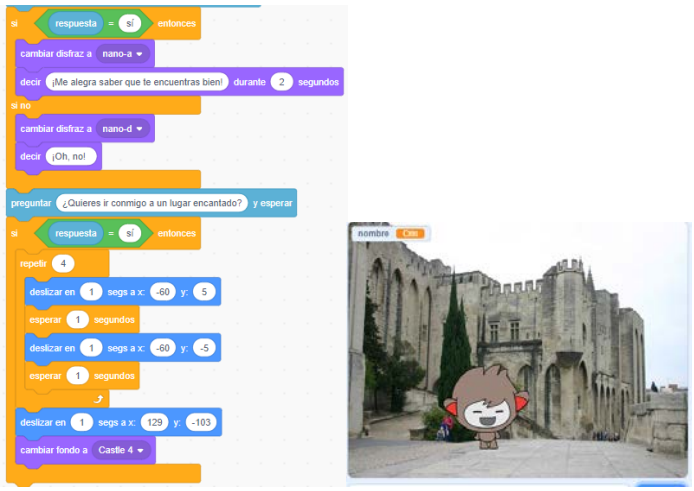


Ahora puedes programar a tu viajero para que cambie de ubicación si le añades el siguiente código:

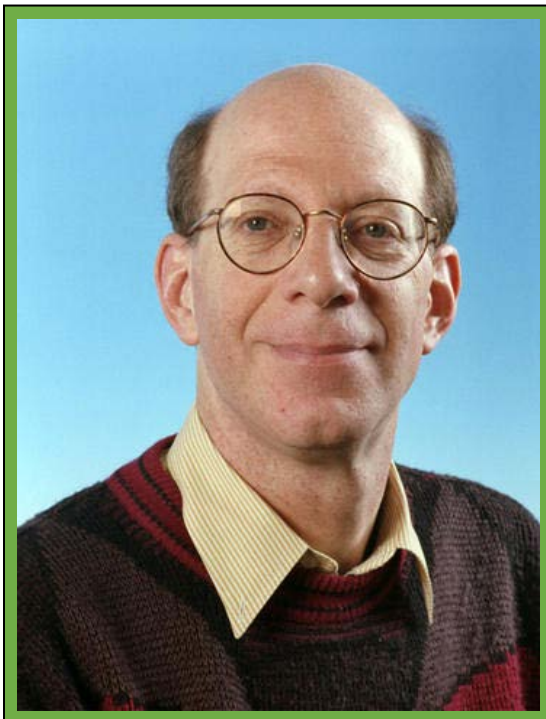
Asegúrate que tu viajero este en el escenario anterior cuando empieza a responder.

Añade este bloque al principio del código para que suceda lo antes mencionado:

Finalmente, si quieres hacer que tu viajero se vea contento y animado, intenta agregar este código dentro de tu bloque "**si**" para hacer que salte 4 veces si la respuesta es sí:



Nota: la respuesta deberá ser igual a la que escribiste en el código, particularmente cuando respondes sí. Scratch no distingue entre mayúsculas y minúsculas, ni tampoco entre palabras acentuadas.



Andrew Stuart "Andy" Tanenbaum (nacidoo el 16 de marzo de 1944), es profesor de ciencias de la computación de la Universidad Libre de Ámsterdam, Países Bajos.

Tanenbaum es mejor conocido por ser el creador de "MINIX", un sistema tipo "UNIX" gratuito con propósitos educativos, y por sus libros sobre ciencias de la computación.

Minix, fuen el sistema operativo que inspiró la creación del sistema operativo "LINUX". En 1992, Tanenbaum participó en Usenet en un encendido debate con Linus Torvalds, el "creador" de Linux, sobre los méritos de la idea de Linus de utilizar un núcleo monolítico en lugar de micro núcleos.

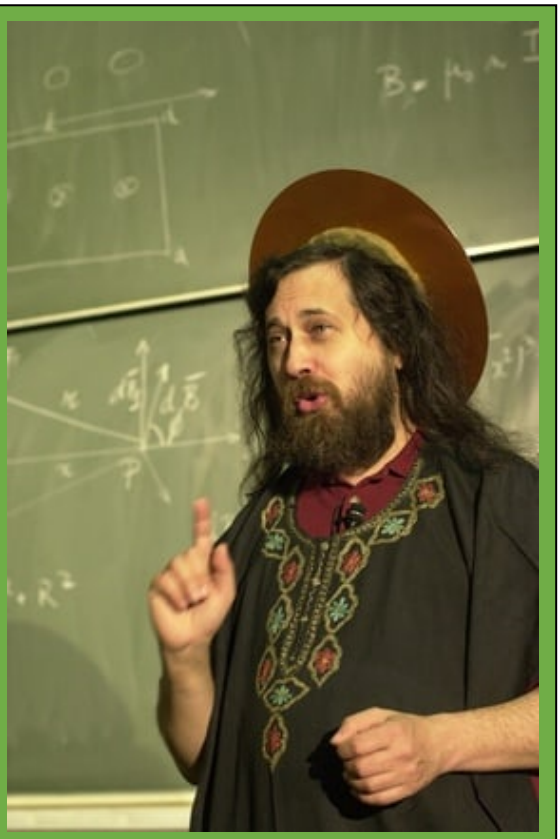
Tanenbaum es autor, junto a otros miembros de la Universidad Libre de Ámsterdam, del sistema operativo distribuido de investigación "AMOEBA", basado en una arquitectura de micro núcleo. Tanenbaum también es el creador de "GLOBE", un software que provee la infraestructura para un sistema distribuido a nivel mundial.

Richard Matthew Stallman (nace el 16 de marzo de 1953 en Manhattan, Nueva York): Es un programador, activista e idealista estadounidense; fundador del movimiento del "Software Libre", del Proyecto "GNU" y de la "Fundación para el Software Libre" (FSF).

Entre sus logros destacados como programador se incluye la realización del editor de texto "GNU Emacs", el compilador "GCC", el depurador "GDB", y el lenguaje de construcción "GNU Make"; todos bajo la rúbrica del Proyecto GNU. Adicionalmente, es Ampliamente conocido por el establecimiento de un marco de referencia moral, político y legal para el software libre.

Cabe destacar que el Linux que conoces (al igual que sus diversas distribuciones), en realidad debe llamarse GNU/LINUX. Dado que Linus Torvaldas contribuyo únicamente en la construcción del núcleo, inmerso en el Sistema GNU de Stallman y la FSF.

En la actualidad, el código del GNU/LINUX ha sido modificado y mejorado por muchísimos entusiastas, quedando cerca de un 10% como aportación de Stallman y poco más del 1% como aportación de Linus Torvalds.



11. Juego Mental

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Identifica operaciones matemáticas, y las relaciona con sus respectivas variables. Utiliza las funciones aprendidas para crear un reto al participante.	Por medio de un IDE: Scratch	Crea un personaje. Crea dos variables. Utiliza números aleatorios para jugar. Crea un cronómetro. Crea un botón para iniciar el juego.	Programa un juego interactivo relacionado con habilidad numérica.

11. Juego Mental

Introducción

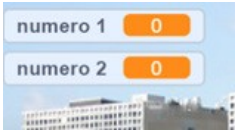
En este proyecto crearás un juego de preguntas sobre las tablas de multiplicar, en el que tendrás que conseguir tantas respuestas correctas como puedas en 30 segundos.

Crea un personaje

Empieza un nuevo proyecto de Scratch. Escoge un personaje y un fondo para tu juego. ¡Puedes escoger el que más te guste! Aquí tienes un ejemplo.

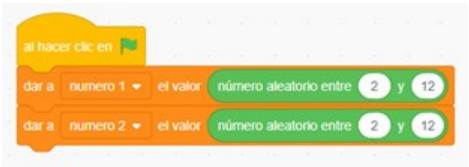
Crea dos Variables

Crea dos nuevas variables llamadas “número 1” y “número 2”. Estas variables almacenarán los 2 números que se van a multiplicar.

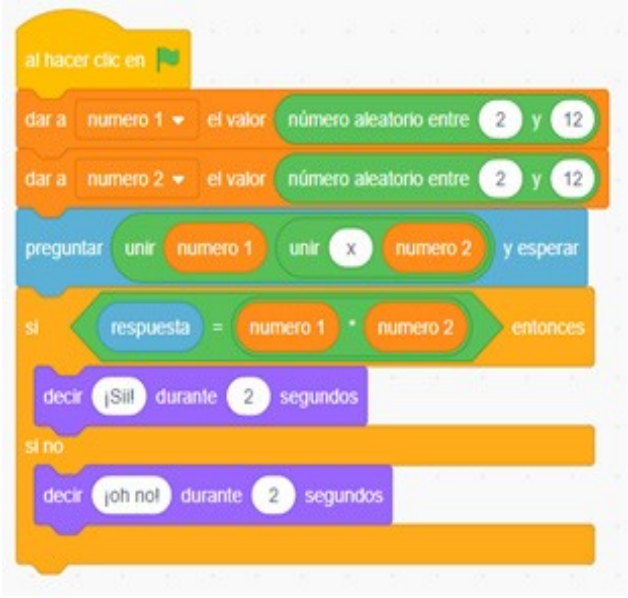


Utiliza Números Aleatorios

Añade código a tu personaje, para fijar dos variables a un número aleatorio entre 2 y 12. Un número aleatorio es como un número al azar, similar a cuando lanzas una moneda o un dado; es decir, no sabes el valor que tendrá hasta que el evento ocurre.



A continuación puedes pedir al jugador que dé una respuesta y contestarle si es correcta o incorrecta.



Prueba tu proyecto del todo, dando algunas respuestas correctas y otras incorrectas.

Añade un bucle “**por siempre**” alrededor de este código, para que se le hagan un montón de preguntas al jugador, hasta que se canse.

Crea un Cronómetro

Ahora crea un cronómetro de cuenta atrás en el escenario, usando una variable que se llame tiempo.

Vuelve a probar tu proyecto. Debería de hacer preguntas hasta que se agote el tiempo.

Reto:

Cambia los disfraces de tu personaje dependiendo de la respuesta del jugador.

Reto:

Sería muy útil añadir puntuación a tu juego. Suma un punto por cada respuesta correcta. O también podrías volver a dejar la puntuación del jugador en 0 si se equivoca en la respuesta.

Crea un Botón para iniciar el juego

Estaría bien también añadir un botón de "jugar", para que puedas intentarlo varias veces.

Para ello, crea un nuevo objeto con un botón "jugar". El jugador hará clic para empezar un juego nuevo. Puedes dibujarlo o bien, editar un objeto de la biblioteca de Scratch.

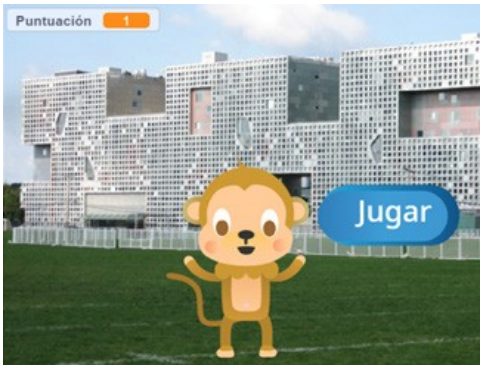
Añade este código a tu nuevo botón:

Este código muestra el botón de jugar cuando el proyecto empieza. Al presionar el botón, éste se esconde y envía un mensaje que dará inicio al juego.

Necesitarás editar el código de tu personaje para que el juego empiece cuando recibe el mensaje de inicio y no cuando se presione la bandera.

Para lograrlo, sustituye el código **al presionar bandera verde**, por **al recibir mensaje1**.

Ahora, haz clic en la bandera verde y a continuación presiona tu nuevo botón de jugar para probarlo. Al empezar no deberías de ver el juego, hasta que hayas presionado el botón.



Reto:

¡Agrega una pantalla de inicio!

¿Puedes añadir otro fondo a tu escenario?

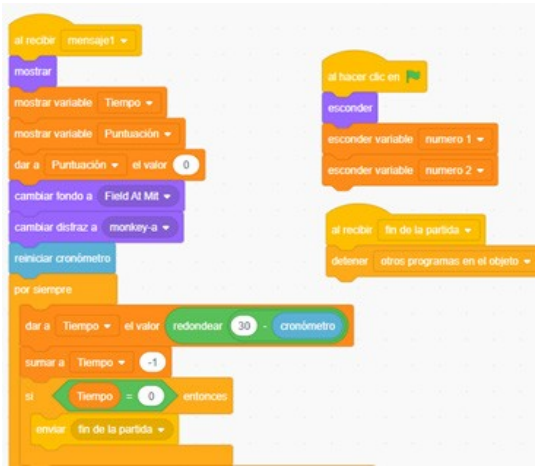
Ayuda:

Puedes usar los bloques al **recibir empezar** y al **recibir final** para cambiar los fondos.

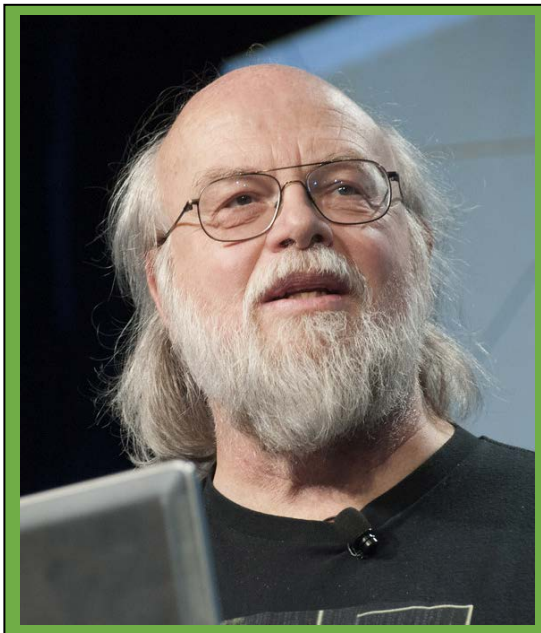
Para **mostrar** o **esconder** al personaje cuando cambie el fondo del juego puedes usar los bloques **mostrar** y **esconder**.

Para mostrar o esconder el cronómetro y la puntuación cuando cambien los fondos puedes usar los bloques **mostrar variable** y **esconder variable**.

Tu código quedará muy parecido a este:



El código de tu botón debe quedar parecido a estos bloques:



James Gosling (19 de mayo de 1955): Es un famoso científico de la computación; mientras trabajaba para su doctorado, escribió una versión de Emacs (Gosling Emacs), y antes de unirse a Sun Microsystems, construyó una versión multi-procesador de Unix, así como varios compiladores y sistemas de correo.

Desde 1984 James Gosling ha trabajado en la compañía estadounidense Sun Microsystems donde fue vicepresidente hasta que ésta fue comprada por Oracle.

Posteriormente, empezó a trabajar en Google el 28 de marzo de 2011, anunciándolo en una entrada en su blog. Actualmente es Jefe de Arquitectura de Software en la empresa Liquid Robotics.

Gosling es reconocido como el creador del lenguaje de programación "JAVA". Realizó el diseño original y la implementación del compilador original y la máquina virtual Java. Además, Gosling ha contribuido con otros proyectos de software como "NeWS" y "Gosling Emacs".

En el 2015 recibió la medalla John von Neumann de la IEEE por sus contribuciones al desarrollo informático.

12. Guerra de galaxias

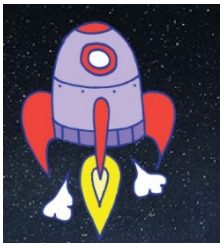
Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Realiza un juego, en el cual el jugador controla una nave espacial que dispara relámpagos a los enemigos.	Por medio de un IDE: Scratch	Construye una nave espacial. Genera un rayo para defenderte. Crea hipopótamos espaciales voladores. Utiliza variables para crear puntajes.	Crea un conjunto de órdenes para generar un programa. Abstrae y refuerza el uso de variables.

12. Guerra de Galaxias

Construye una nave espacial

Vas a construir una nave espacial que defienda a la Tierra de unas criaturas voladoras. Empieza por seleccionar el fondo de “**estrellas**” y el objeto de la “**nave espacial**”.



Reduce el tamaño de la nave espacial y muévela cerca de la parte inferior de la pantalla.

Luego añade el código para mover la nave espacial a la izquierda cuando se pulse la tecla de flecha izquierda. Utiliza para ello los siguientes bloques:

Ahora necesitas agregar un código para mover la nave espacial a la derecha cuando se pulsa la tecla de flecha derecha. Es similar al anterior, inténtalo.

Genera Rayos para defenderte

Una nave que no dispara no nos puede defender de nadie, así que ahora harás que la nave espacial pueda disparar rayos.

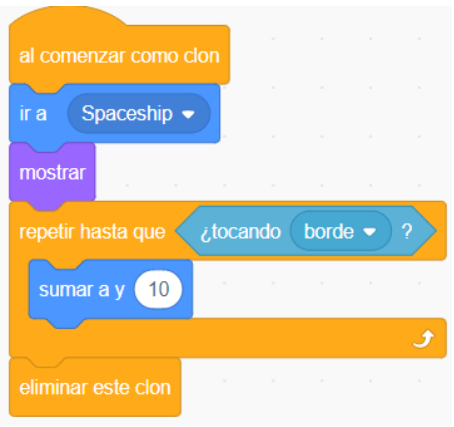
De modo que tendrás que añadir el objeto que simbolizará nuestro “rayo’” de la biblioteca Scratch. Haz clic en el disfraz del objeto y gira el “rayo” hasta que se vea como en la siguiente imagen:

Cuando se inicia el juego, los rayos deben estar ocultos hasta que la nave espacial dispare. Así que oculta tu rayo (aún no lo has disparado).

Un solo rayo no es suficiente para defender a la Tierra. Así que añade el siguiente código a la nave espacial para crear un nuevo rayo cada vez que se pulsa la tecla de espacio.

Con estos bloques siempre creas un nuevo “clon” de tu “rayo”, este debe comenzar en el mismo lugar que la nave espacial y luego subir por el escenario hasta que toque el borde.

Añade lo siguiente para tu “rayo”.

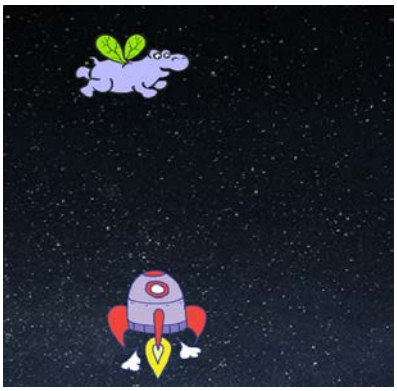


Reto:

¿Puedes añadir puntuación a tu juego?

Hipopótamos Espaciales Voladores

Añade un montón de hipopótamos voladores que están tratando de destruir tu nave espacial. Para esto, crea un objeto nuevo desde la imagen hippo1 en la biblioteca Scratch.



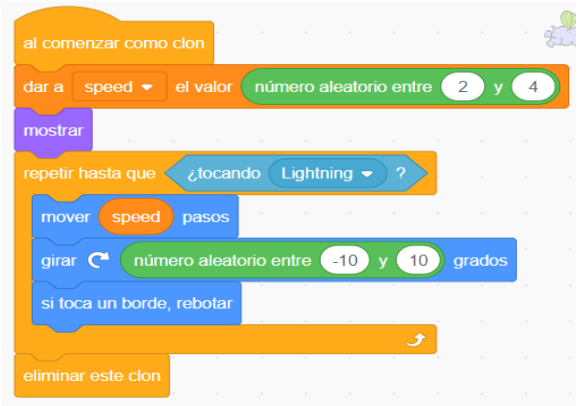
Establece su estilo de rotación para girar a la derecha solamente y agrega el siguiente código para ocultar el objeto cuando se inicia el juego:

Crea una nueva variable llamada velocidad solo para el objeto hipopótamo.

El siguiente código creará un nuevo hipopótamo en pocos segundos (tiempo elegido al azar).

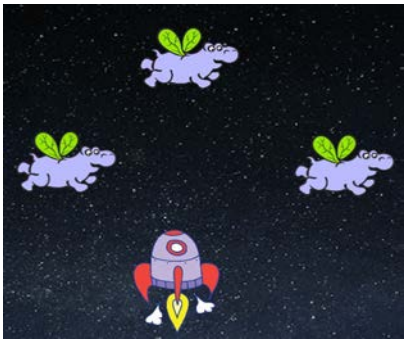
Cuando inicie cada clon de hipopótamo, haz que se mueva por el escenario (a una velocidad aleatoria) hasta que sea golpeado por el rayo.

Añade el siguiente código para el objeto hipopótamo:



Corrobora el código hipopótamo.

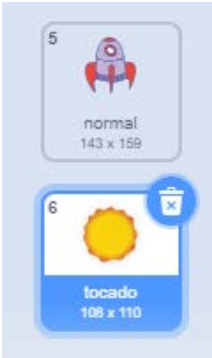
Se debe ver un nuevo clon hipopótamo que aparece a los pocos segundos. Cada uno se mueve a su propio ritmo.



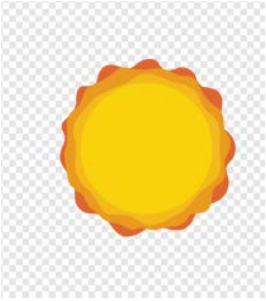
Pon a prueba el cañón láser. ¿Qué pasa cuando acierta sobre un hipopótamo?, ¿se desvanece?

Cuando un hipopótamo toca tu nave espacial, ¡Tienes que hacer que la nave espacial explote!

Para ello, primero hay que asegurarse de que la nave espacial tiene 2 disfraces llamados “normal” y “tocado”.



El disfraz “tocado” de la nave espacial se puede hacer mediante la importación de la imagen “Sun”, de la biblioteca Scratch, y el uso de la herramienta “Colorear una forma’” para cambiar su color.



Añade este código a la nave espacial para que se cambie de disfraz cada vez que choque con un hipopótamo volador:

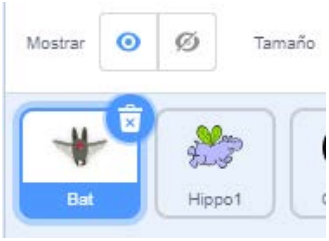
¿Te diste cuenta de que se ha transmitido un mensaje de “tocado” (mensaje 1) en el código anterior? Se puede utilizar este mensaje para hacer que todos los hipopótamos desaparezcan cuando alcance la nave espacial.

Añade este código al hipopótamo:

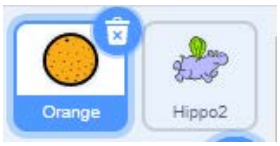
Haz este juego más interesante aún. Ahora haz un murciélago que lance naranjas a la nave espacial.

En primer lugar, haz un nuevo objeto murciélago que se moverá en la parte superior del escenario por siempre.

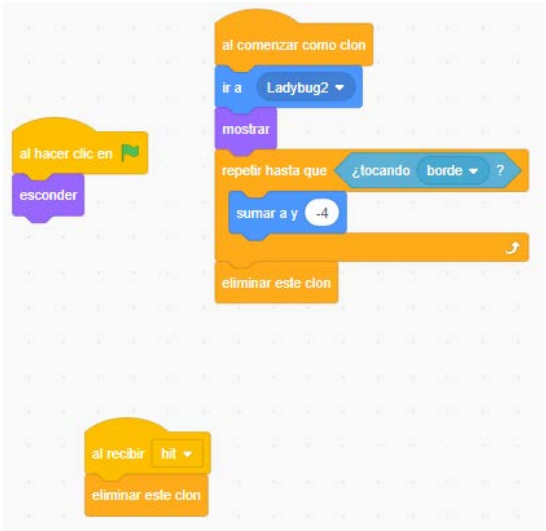
Añade un nuevo objeto “Bat” de la biblioteca de Scratch.



Crea un nuevo objeto ‘Naranja’ de la biblioteca de Scratch.



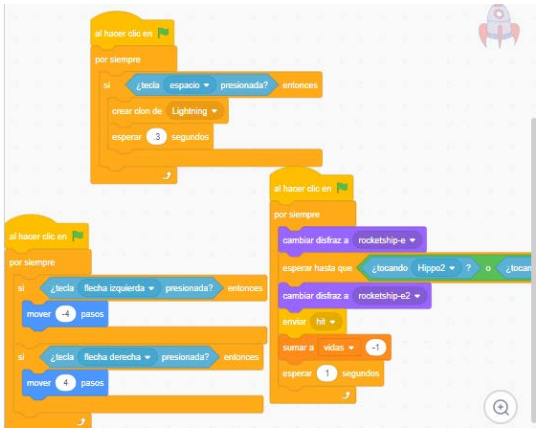
Añade este código para que cada clon de “Naranja” caiga por el escenario desde el murciélago hacia la nave espacial:



En el objeto nave espacial, modifica el código para que se destruya si se toca un hipopótamo al igual que si se toca una naranja:

Tu código quedará al final así:

Código para la nave espacial

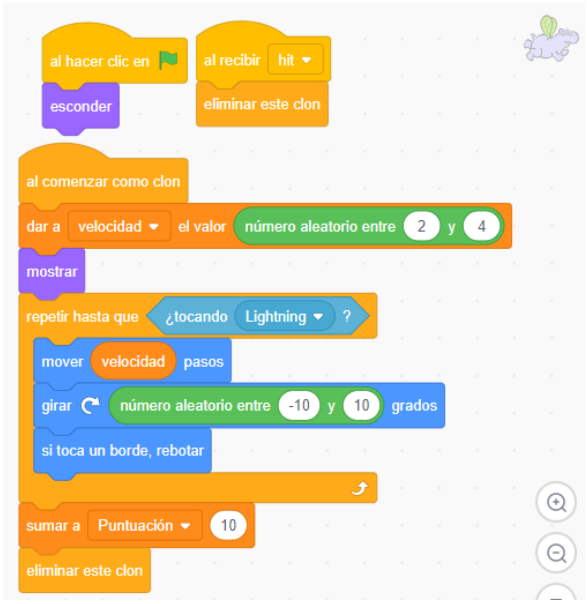


Código para el rayo:

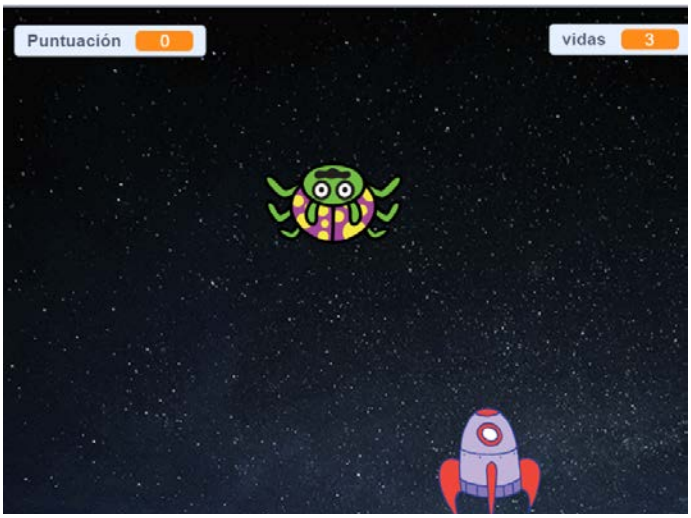
Código para el murciélago:

Código para la naranja:

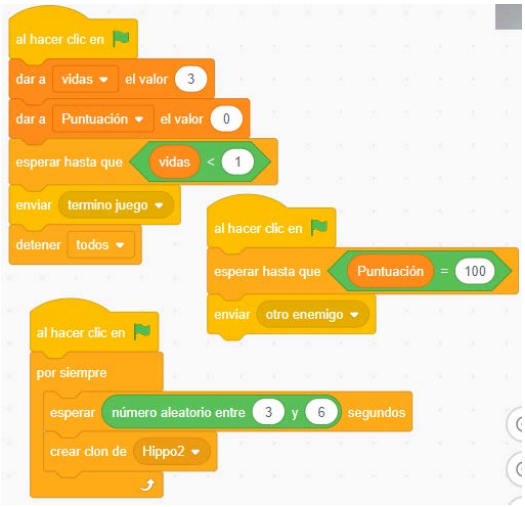
Código para el hipopótamo:



Ve el penúltimo bloque; ese es el código para la puntuación y en tu pantalla se verá de la siguiente manera.

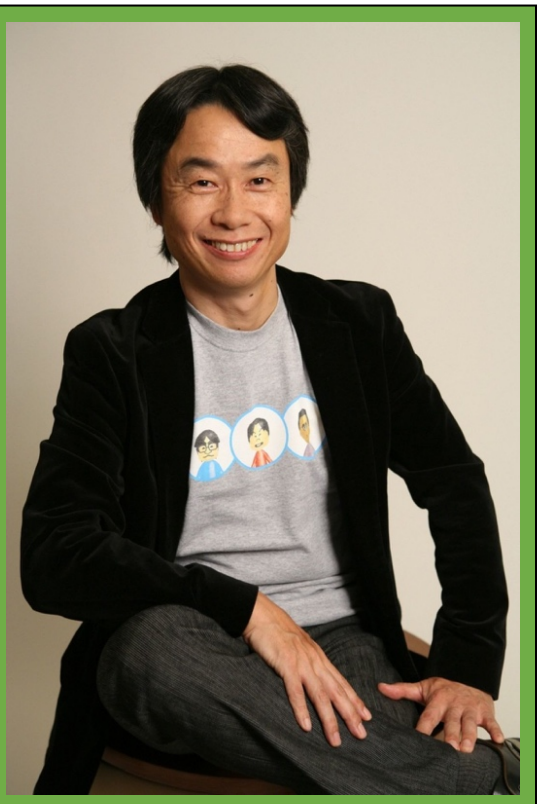


Código en el escenario:



Shigeru Miyamoto (nace en Sonobe, Kioto, Japón, 16 de noviembre de 1952): Es un diseñador y productor de videojuegos japonés que trabaja para Nintendo desde 1977.

Es considerado como «el padre de los videojuegos modernos» o «el Walt Disney de los juegos electrónicos» por haber creado algunas de las franquicias más influyentes de la industria, entre las que se encuentran Mario, Donkey Kong, The Legend of Zelda, Star Fox, Pikmin y F-Zero.



Takashi Tezuka (nace el 17 de noviembre de 1960): Diseñador y director de videojuegos de Nintendo. Sus trabajos se remontan hasta el videojuego Super Mario Bros original donde tomó el rol de asistente de dirección del proyecto.

Tezuka se convirtió en responsable del departamento Nintendo Entertainment Analysis and Development junto a Shigeru Miyamoto, responsable de las sagas de Mario y The Legend of Zelda, entre otras. En 2005 ascendió a Productor general del departamento, gracias a la reestructuración interna de Nintendo, supervisando la mayoría de títulos creados dentro de la compañía. A pesar de su cargo, aún se ocupa de algunos diseños, como los realizados para New Super Mario Bros.

13. Carrera de Coches

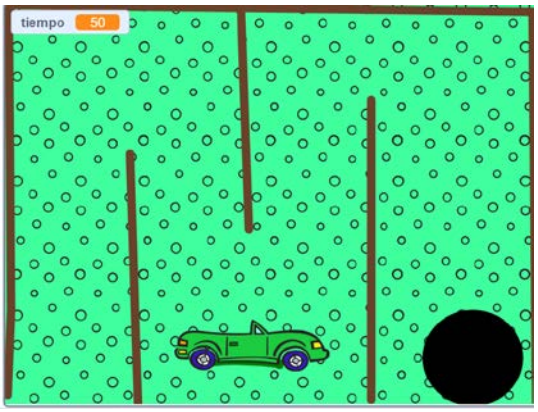
Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Utiliza las funciones de dibujo, espacio y medida para elaborar un juego.	Por medio de un IDE: Scratch	Prepara un camino para el coche. Realiza un choque con disfraces. Crea un camino con obstáculos.	Programa un juego en el que interviene forma, medida y condiciones lógicas.

13. Carrera de Coches

Introducción

Ésta es la última práctica del módulo 3; en ella, harás que un coche tenga que sortear caminos para alcanzar la meta. ¡Comencemos!

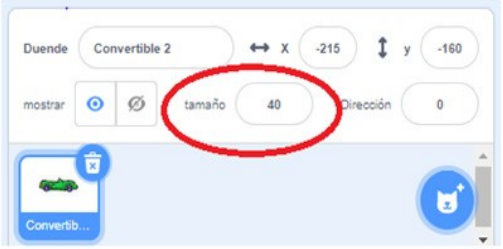


Selecciona los objetos.

Primero, crea un nuevo proyecto en Scratch, elimina el objeto del gato y selecciona el objeto “convertible”; que es la animación de un automóvil. Puedes elegir entre las dos opciones del ejemplo.

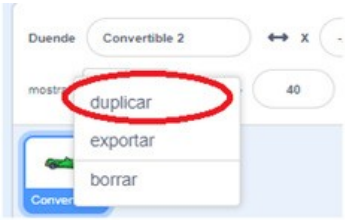
Modifica el tamaño del objeto.

Tendrás que cambiar el tamaño del objeto para moverlo por el camino. Un tamaño que te puede servir es 40, por ejemplo.

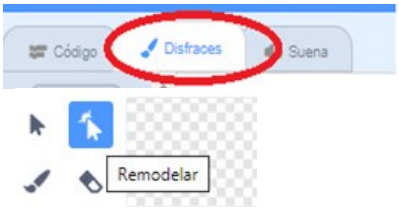


Prepara los objetos para realizar el efecto de un choque.

Ahora, duplica el objeto para intercambiarlo por el primero cuando sea necesario en el juego. En específico, cuando el carrito toque el extremo del camino, debes mostrarlo como un carrito descompuesto o chocado.



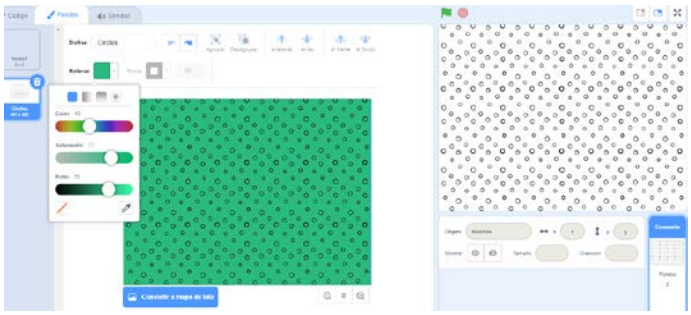
Después de duplicar el objeto deberás modificarlo. Usa la pestaña: “Disfraces”. Encontrarás la herramienta: Remodelar



Una vez que seleccionaste la opción Remodelar, elige un fragmento del objeto y muévelo para deformarlo. ¡Acabes de crear un carrito chocado!

Prepara tu camino

Ahora debes preparar el escenario done tu carrito transitará. Agrega un fondo que puedas modificar como más te guste.



Ahora agrega las siguientes características a tu escenario:

Dibuja el camino que deberá recorrer tu carrito.

Dibuja algunos obstáculos; esto hará que sea más difícil el recorrido de tu carrito.

Las fronteras de tu camino deberá tener el color que elijas para determinar que chocó cuando tu carrito lo toque

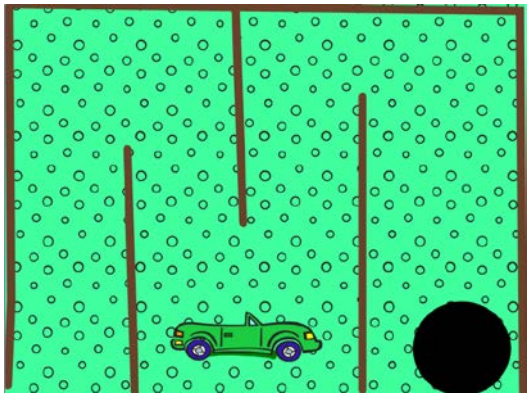
También deberás seleccionar un color que indique el final de tu camino (la meta)

Para seleccionar estos colores puedes definir los valores de saturación, brillo y color como se ve en la siguiente imagen:

Revisa que los valores de saturación, brillo y color coincidan, de lo contrario, no funcionará el código.

En la imagen anterior observarás los valores de saturación, brillo y color para dibujar el borde del camino y el final del camino, respectivamente. En este caso se seleccionaron variantes de café y negro.

Esta es la imagen final del camino terminado con los dos colores.



El código para tu juego va a ser muy simple y requerirá los siguientes componentes:

El bloque **al hacer clic en "bandera"**, para ejecutar el código**.**

Un bloque **mostrar variable**. Antes de utilizar este bloque, deberás crear una variable con el nombre **tiempo**. Esta variable registrará el tiempo que tarda en llegar el carrito al final del camino.

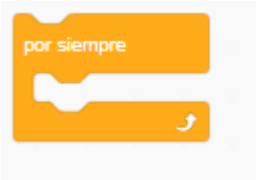
El bloque **cambiar disfraz** que usaras para cambiar el carrito original por el que deformaste, para simular un choque cuando se requiera.

Los bloques **apuntar en dirección**, e **ir a x:** -190 **y:** -150; determinarán la posición y orientación del objeto.

Los siguientes bloques de código te servirán para calcular el tiempo que tarde en llegar el carrito al final del camino.

El bloque **dar a tiempo el valor 0** iniciará la variable "tiempo" en cero.

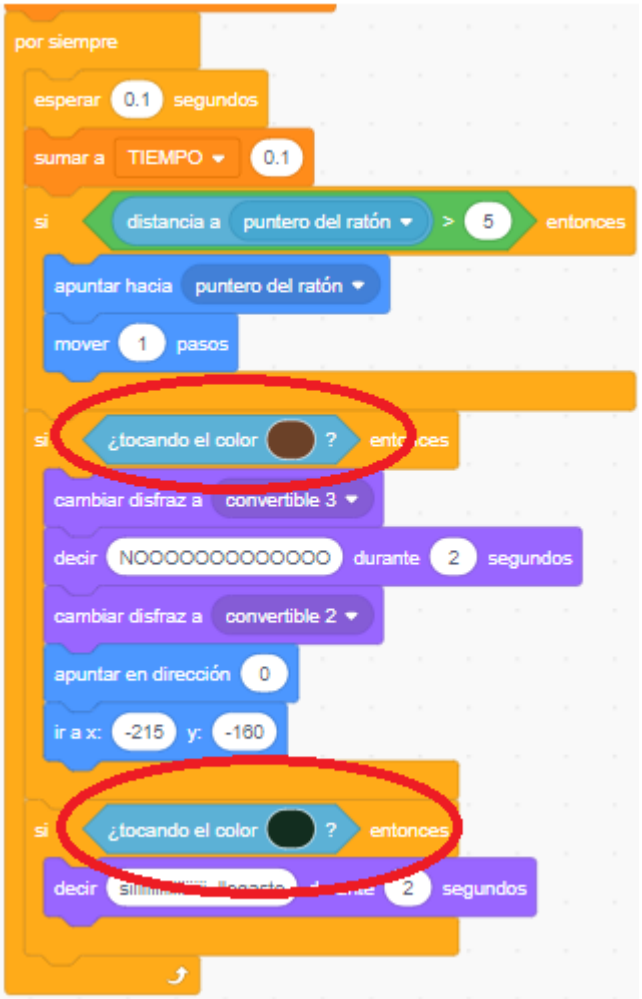
Ocuparás también un bloque “**por siempre”** y dentro de éste ingresarás el resto del código.



Lo primero que colocaras dentro del bloque “**por siempre”** serán los siguientes bloques de código.

Con ellos, indicarás que tus acciones se van a actualizar cada 0.1 segundos y se actualizará también la variable tiempo.

Los siguientes bloques de código que introducirás, determinarán los movimientos y efectos del carrito cuando toque los extremos o llegue al final del camino. Recuerda que es muy importante que los valores de saturación, brillo y color sean los mismos que se indican en el círculo de color rojo que se muestra en la siguiente imagen de código, de lo contrario no funcionará el juego.



¡Explora haciendo todas las modificaciones que quieras y diviértete!

Margaret Hamilton (17 de agosto de 1936): Es una científica computacional, matemática e ingeniera de sistemas.

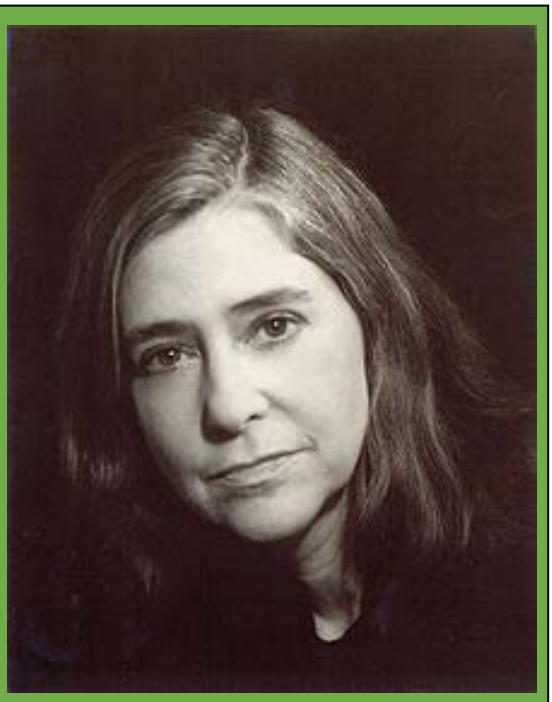
Fue directora de la División de Ingeniería de Software del Laboratorio de Instrumentación del MIT; donde con su equipo desarrolló el software de navegación "a bordo" para el Programa Espacial Apolo.

En 1986, se convirtió en la fundadora y CEO de Hamilton Technologies, Inc. en Cambridge, Massachusetts. La compañía se desarrolló alrededor de un lenguaje universal de sistemas basado en su paradigma de "desarrollo antes del hecho" (DBTF, del inglés Development Before the Fact) para sistemas de diseño de software.

Acuño el término "Ingeniería de Software" para distinguir entre el trabajo de hardware y otras ingenierías. A pesar de que su idea no fue bien recibida al inicio, eventualmente la Ingeniería de Software generó el mismo respeto que otras disciplinas de la ingeniería.

El 22 de noviembre de 2016, Hamilton recibió la Medalla Presidencial de la Libertad, entregada por el expresidente de Estados Unidos Barack Obama, por su trabajo en la NASA durante las misiones Apolo.

Su enfoque del diseño de software fue pionero para la época, ya que los requisitos de robustez y tolerancia a fallos eran una prioridad para la seguridad y supervivencia de los astronautas durante los viajes a la Luna. Es por ello que, para muchos, Margaret Hamilton es considerada una de las primeras personas en convertirse en Ingeniera de Confiabilidad del Sitio (del inglés Site Reliability Engineer).



GOBIERNO DE LA CIUDAD DE MÉXICO

SECRETARÍA DE EDUCACIÓN, CIENCIA, TECNOLOGÍA E INNOVACIÓN

Raspberry Pi

AEFCM
AUTORIDAD EDUCATIVA FEDERAL EN LA CIUDAD DE MÉXICO

Raspbian

python™

scratch

Este material forma parte de una iniciativa que pretende generar en niños y jóvenes, de una manera lúdica, el gusto por la programación y el desarrollo de tecnología con base en herramientas abiertas y de bajo costo. Estamos convencidos de que esta estrategia les brindará, como agradable efecto secundario, una estructura de pensamiento lógico que les preparará para desarrollarse en el campo de las ciencias, la matemática y la ingeniería.

Algunas de las prácticas de este documento fueron inspiradas en la red de clubes de código de la “Fundación Raspberry Pi” y en la iniciativa “Programo Ergo Sum”.

https://projects.raspberrypi.org/en

https://www.programoergosum.es