



Cursos gratuitos de programación para
estudiantes de bachillerato

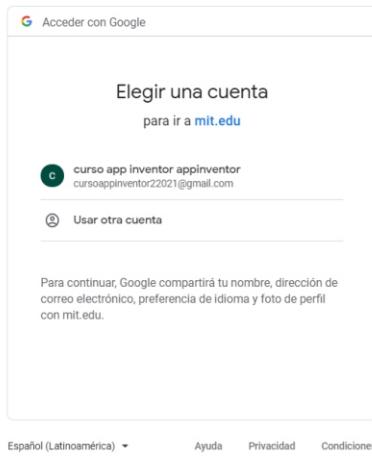
NIVEL 1:
PROGRAMACIÓN MÓVIL
CON APPINVENTOR

Curso de Mit App inventor 2

Conociendo el entorno de trabajo

I. ¿Cómo ingresar a App inventor?

Para comenzar a usar App inventor debemos dirigirnos a la página <http://ai2.appinventor.mit.edu/>
Es posible que nos pida iniciar sesión en una cuenta de google para continuar, si es así, verás esta pantalla:

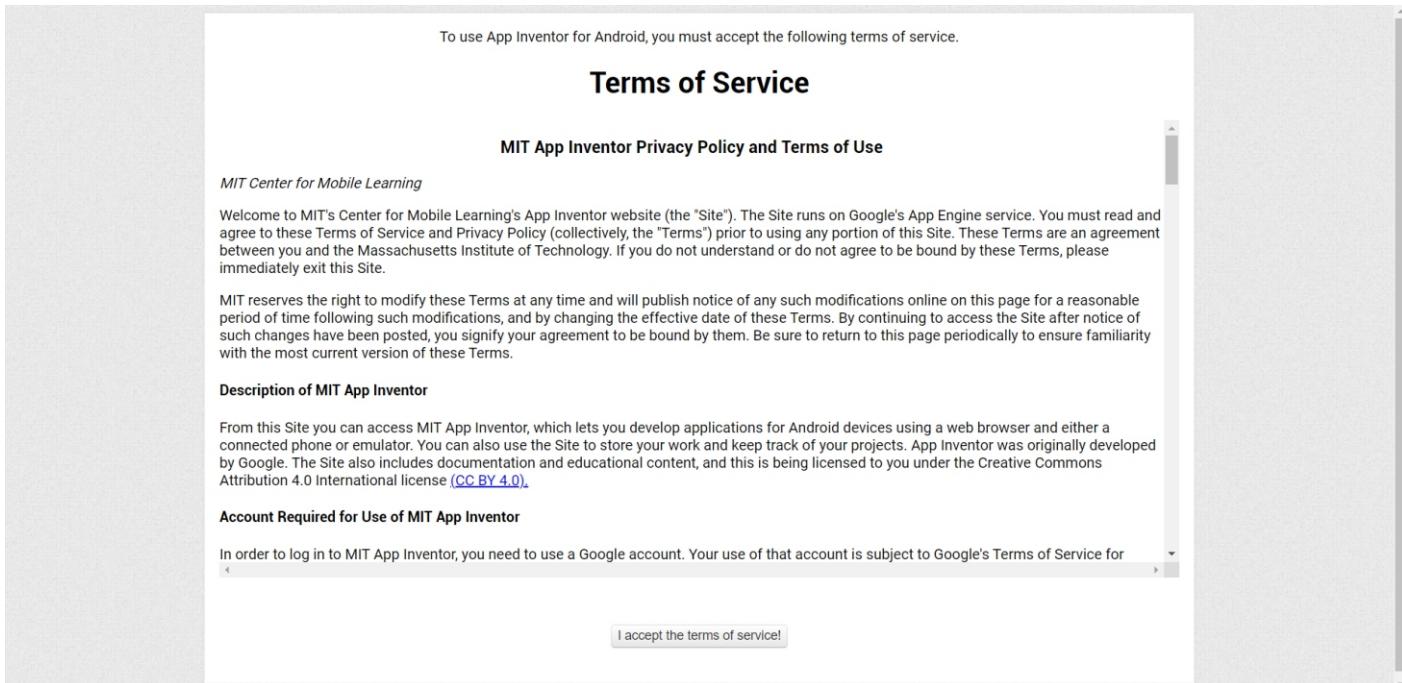


O una como esta:



Inicia sesión con tu cuenta de correo de gmail y contraseña para continuar, no te preocupes, esta página es confiable y segura.

Lo siguiente que vas a ver será esta página donde nos explican los términos de uso de la página, qué hacen con nuestros datos y cuáles son nuestras obligaciones como usuarios, está en inglés pero puedes traducirlo, es importante conocer esta información de seguridad al usar un sitio web.



To use App Inventor for Android, you must accept the following terms of service.

Terms of Service

MIT App Inventor Privacy Policy and Terms of Use

MIT Center for Mobile Learning

Welcome to MIT's Center for Mobile Learning's App Inventor website (the "Site"). The Site runs on Google's App Engine service. You must read and agree to these Terms of Service and Privacy Policy (collectively, the "Terms") prior to using any portion of this Site. These Terms are an agreement between you and the Massachusetts Institute of Technology. If you do not understand or do not agree to be bound by these Terms, please immediately exit this Site.

MIT reserves the right to modify these Terms at any time and will publish notice of any such modifications online on this page for a reasonable period of time following such modifications, and by changing the effective date of these Terms. By continuing to access the Site after notice of such changes have been posted, you signify your agreement to be bound by them. Be sure to return to this page periodically to ensure familiarity with the most current version of these Terms.

Description of MIT App Inventor

From this Site you can access MIT App Inventor, which lets you develop applications for Android devices using a web browser and either a connected phone or emulator. You can also use the Site to store your work and keep track of your projects. App Inventor was originally developed by Google. The Site also includes documentation and educational content, and this is being licensed to you under the Creative Commons Attribution 4.0 International license ([CC BY 4.0](#)).

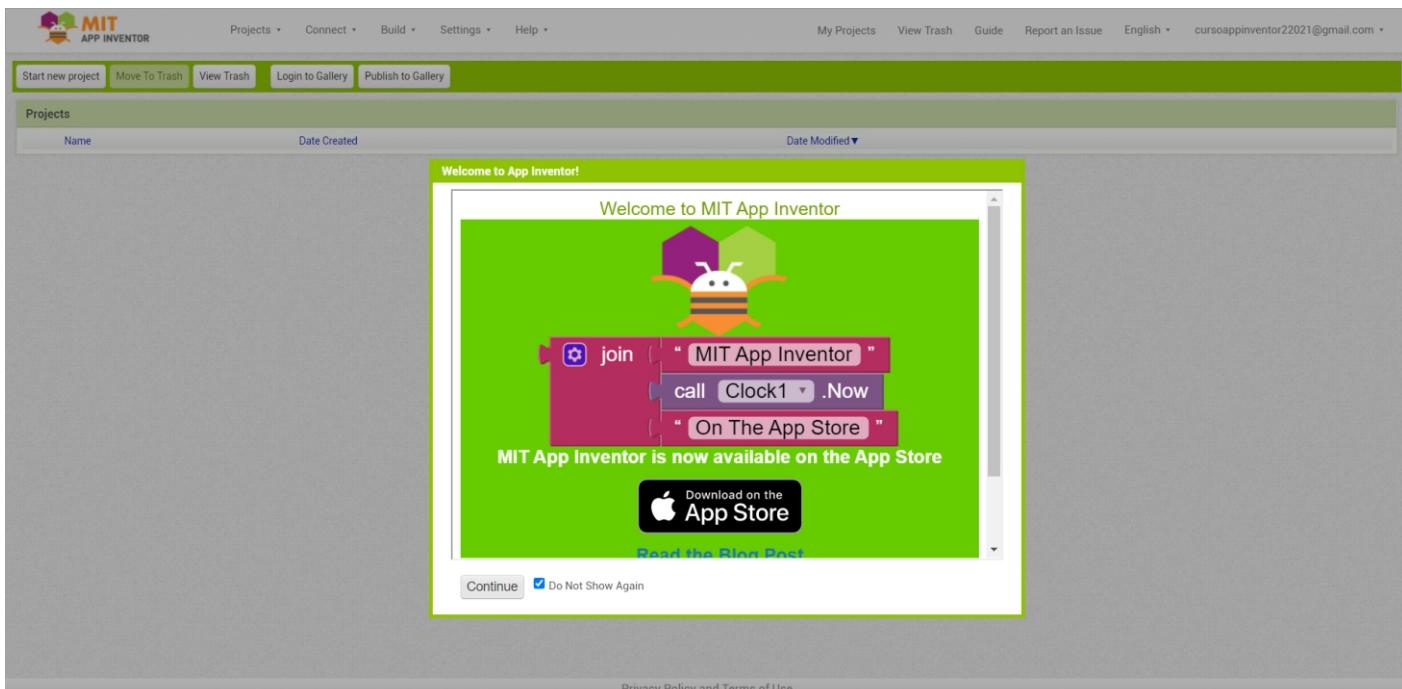
Account Required for Use of MIT App Inventor

In order to log in to MIT App Inventor, you need to use a Google account. Your use of that account is subject to Google's Terms of Service for

I accept the terms of service!

Una vez que hayas leído los términos y condiciones y si estás de acuerdo, da clic en el botón de abajo que dice "Acepto los términos de servicio"

Seguidamente te va a aparecer esta ventana, es un anuncio que te avisa que la app del companion ya está disponible para dispositivos de Apple, marca la opción Do not show again (No mostrar de nuevo) y da clic en Continue.



Welcome to App Inventor!

Welcome to MIT App Inventor

join " MIT App Inventor "

call Clock1 .Now

" On The App Store "

MIT App Inventor is now available on the App Store

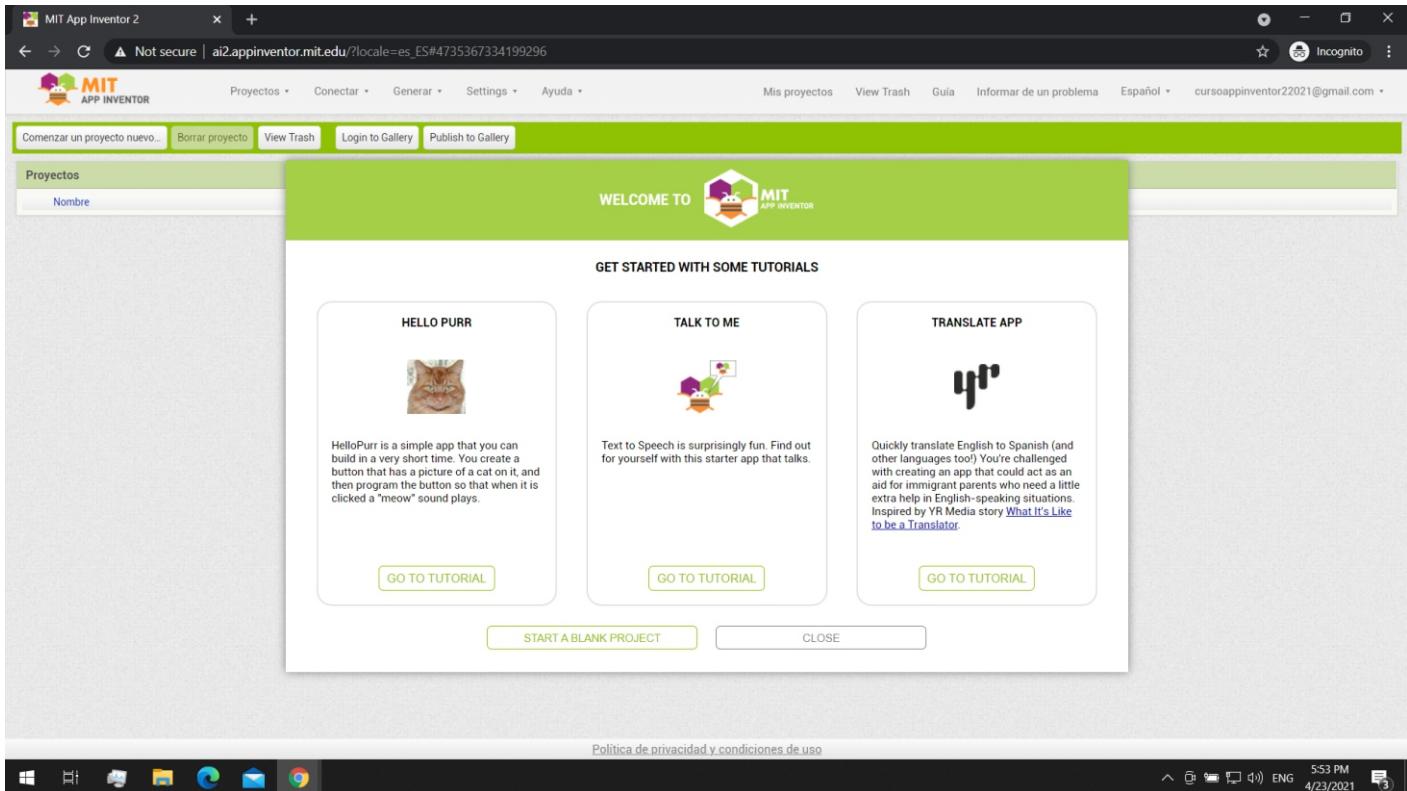
Download on the App Store

Read the Blog Post

Continue Do Not Show Again

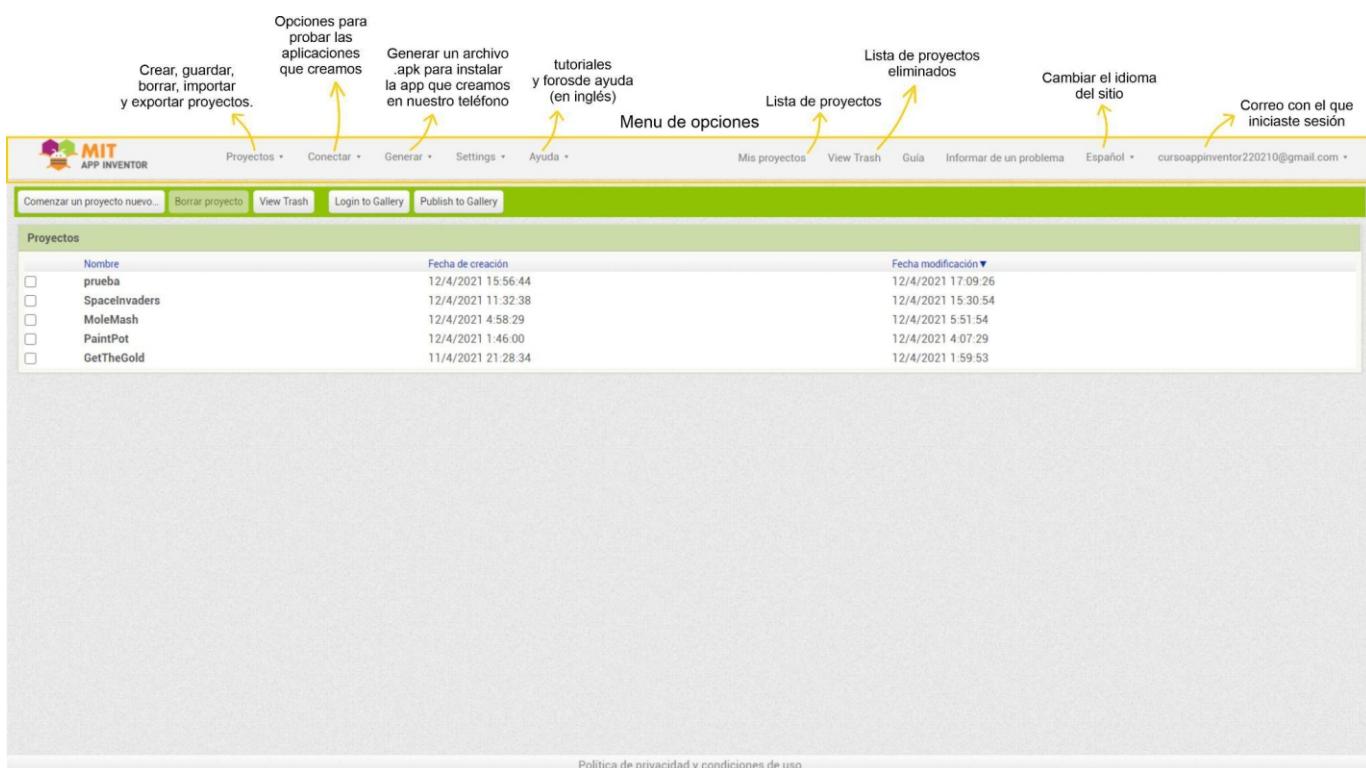
Privacy Policy and Terms of Use

Por último, te va a aparecer esta pantallita que te muestra algunos tutoriales (en inglés), la opción de iniciar un nuevo proyecto (botón verde) y de cerrar (botón negro), vamos a cerrarla.

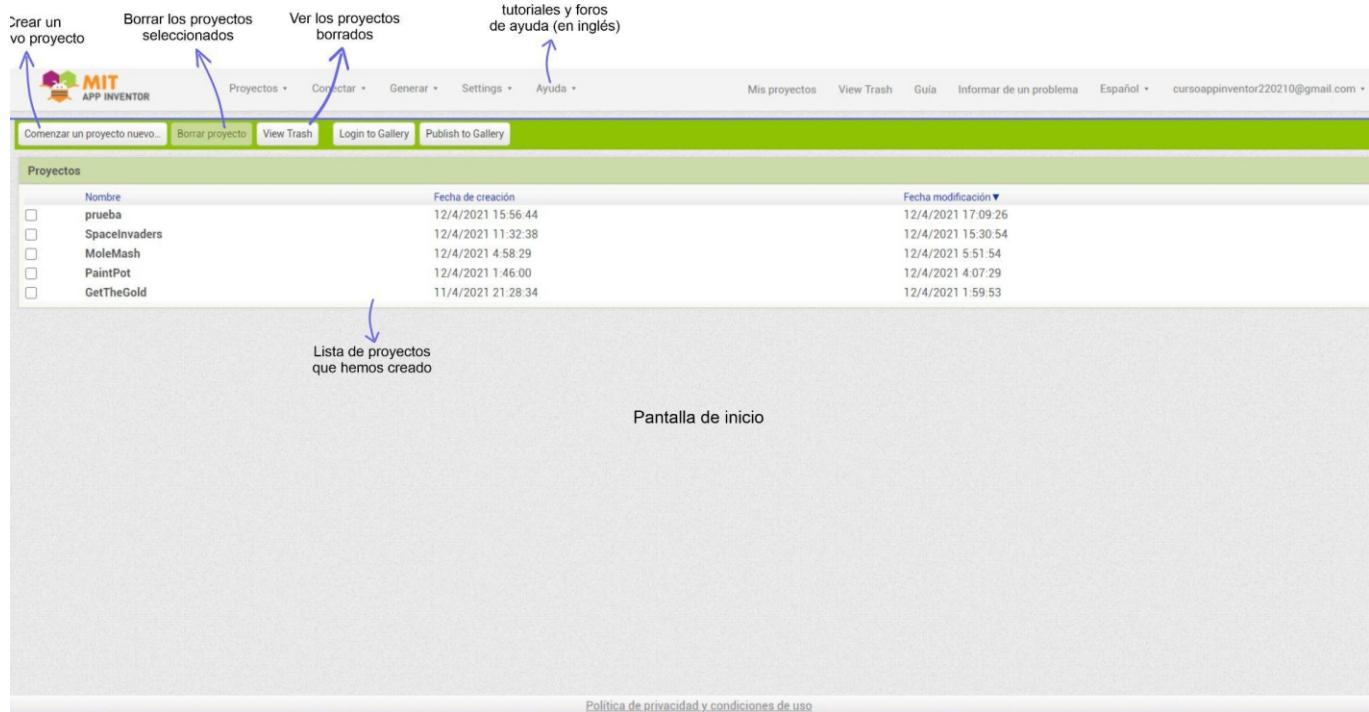


Ahora estás en la pantalla de inicio de App inventor, lo primero que vas a ver será una lista de todos los proyectos que tenés, si es la primera vez que usas app inventor la lista de proyectos estará vacía, ¡pero no por mucho!. En las siguientes imágenes vas a ver una breve descripción de cada parte de esta pantalla.

Menú de navegación:



Pantalla de inicio:



The screenshot shows the main interface of the App Inventor web application. At the top, there's a navigation bar with links like 'Crear un proyecto' (Create a project), 'Borrar los proyectos seleccionados' (Delete selected projects), 'Ver los proyectos borrados' (View deleted projects), 'tutoriales y foros de ayuda (en inglés)' (Tutorials and help forums in English), 'Mis proyectos' (My projects), 'View Trash', 'Guia', 'Informar de un problema', 'Español', and an email address 'cursoappinventor22021@gmail.com'. Below the navigation is a green header bar with buttons for 'Comenzar un proyecto nuevo...', 'Borrar proyecto', 'View Trash', 'Login to Gallery', and 'Publish to Gallery'. The main content area is titled 'Proyectos' (Projects) and displays a table of created projects:

	Nombre	Fecha de creación	Fecha modificación
<input type="checkbox"/>	prueba	12/4/2021 15:56:44	12/4/2021 17:09:26
<input type="checkbox"/>	SpacelInvaders	12/4/2021 11:32:38	12/4/2021 15:30:54
<input type="checkbox"/>	MoleMash	12/4/2021 4:58:29	12/4/2021 5:51:54
<input type="checkbox"/>	PaintPot	12/4/2021 1:46:00	12/4/2021 4:07:29
<input type="checkbox"/>	GetTheGold	11/4/2021 21:28:34	12/4/2021 1:59:53

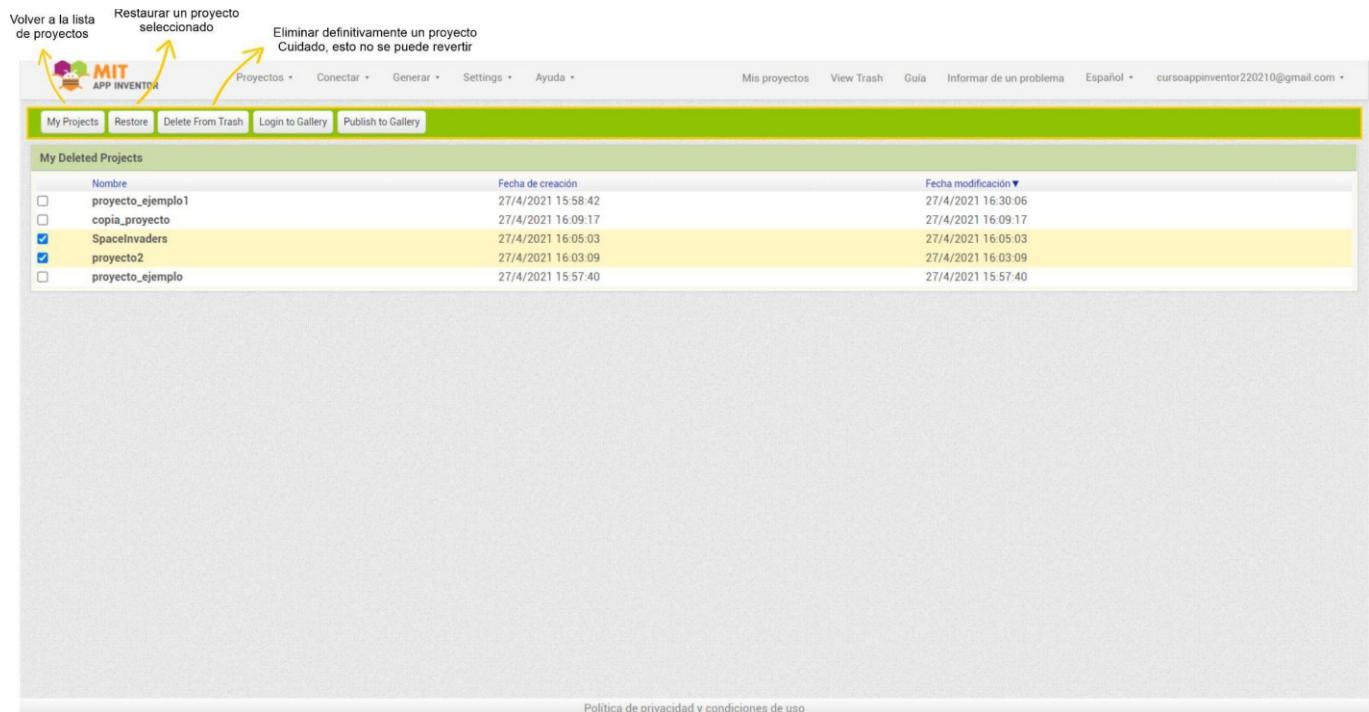
A blue arrow points from the text 'Lista de proyectos que hemos creado' to the table.

Pantalla de inicio

Política de privacidad y condiciones de uso

Basurero:

En la sección View Trash (Ver basura) veremos una lista de los proyectos que hemos borrado, acá podemos recuperarlos o borrarlos definitivamente.



The screenshot shows the 'View Trash' section of the App Inventor interface. At the top, there are buttons for 'Volver a la lista de proyectos' (Return to project list), 'Restaurar un proyecto seleccionado' (Restore selected project), and 'Eliminar definitivamente un proyecto. Cuidado, esto no se puede revertir' (Permanently delete a project. Caution, this cannot be undone). The main content area is titled 'My Deleted Projects' and displays a table of deleted projects:

	Nombre	Fecha de creación	Fecha modificación
<input type="checkbox"/>	projeto_ejemplo1	27/4/2021 15:58:42	27/4/2021 16:30:06
<input type="checkbox"/>	copia_proyecto	27/4/2021 16:09:17	27/4/2021 16:09:17
<input checked="" type="checkbox"/>	SpacelInvaders	27/4/2021 16:05:03	27/4/2021 16:05:03
<input checked="" type="checkbox"/>	projeto2	27/4/2021 16:03:09	27/4/2021 16:03:09
<input type="checkbox"/>	projeto_ejemplo	27/4/2021 15:57:40	27/4/2021 15:57:40

Blue arrows point from the text 'Restaurar un proyecto seleccionado' to the 'Restore' button and from 'Eliminar definitivamente un proyecto' to the 'Delete' button. A yellow box highlights the row for 'SpacelInvaders'.

Política de privacidad y condiciones de uso

II. ¿Cómo crear un proyecto nuevo?

Para crear un proyecto da clic en el botón Comenzar un proyecto nuevo y dale un nombre

Para crear un nuevo proyecto hay que presionar este botón

Nombre: nuevo_proyecto

Cancelar Aceptar

Y nos aparecerá esta ventana donde se debe poner un nombre a al proyecto, Los nombres no deben tener simbolos ni espacios, si son dos palabras se deben separar por guión bajo _

También podés ir a Proyectos > Comenzar un nuevo proyecto como se muestra en la siguiente imagen. Esto abrirá la misma ventana que se muestra arriba para nombrar tu nuevo proyecto.

Nombre: nuevo_proyecto

Cancelar Aceptar

Y nos aparecerá esta ventana donde se debe poner un nombre a al proyecto, Los nombres no deben tener simbolos ni espacios, si son dos palabras se deben separar por guión bajo _

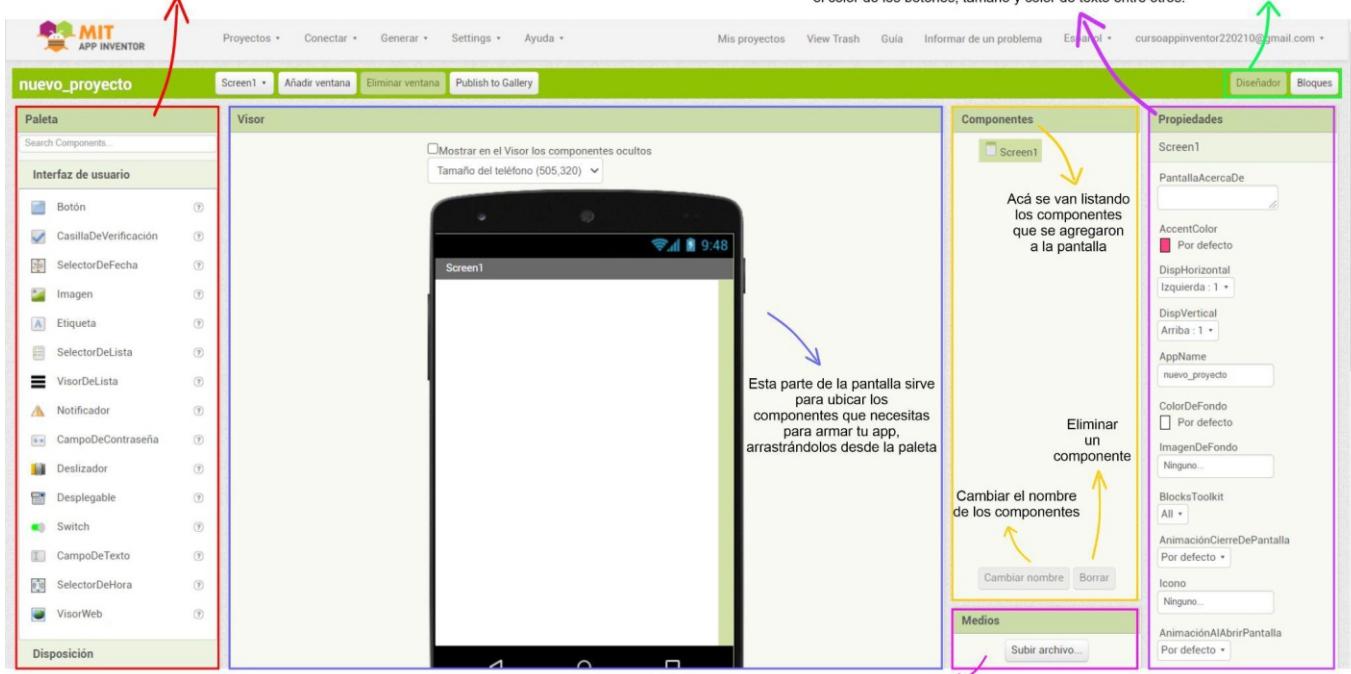
Una vez creado un proyecto vas a ver la pantalla de diseño, en ella se encuentran la paleta que contiene los componentes con los que se construyen la aplicación, la sección de propiedades donde podemos editar aspectos y funcionamiento de los componentes agregados y una pantalla de celular que permite agregar los componentes y ver cómo quedarán en la aplicación.

Diseñador:

La paleta contiene todos los componentes que se pueden agregar a la aplicación como botones, etiquetas, sensores, medios, etc.

Acá se edita el aspecto visual y el funcionamiento de los componentes que se agregaron a la pantalla, como el color de los botones, tamaño y color de texto entre otros.

Botones para cambiar a la vista de bloques o volver al diseñador



Bloques:

Por último, en la sección de bloques se encuentran todos los bloques con los que vamos a programar la aplicación, además de una zona en blanco que es el área de trabajo a donde se deben arrastrar los bloques para programar.

Acá se encuentran los bloques para programar las aplicaciones

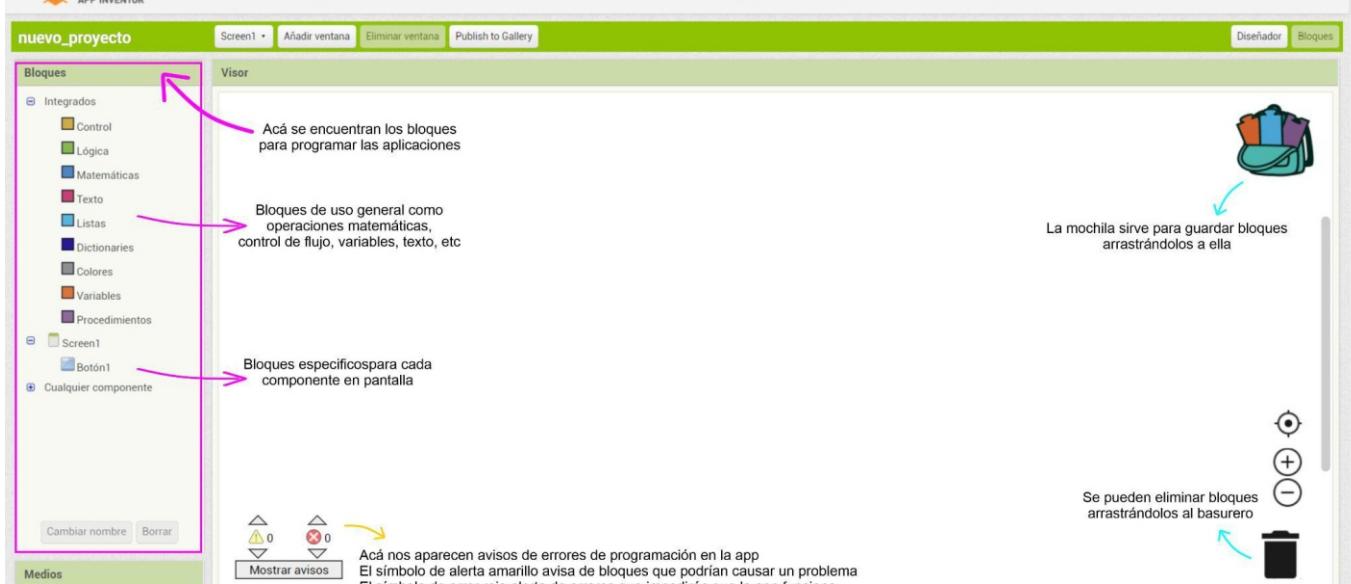
Bloques de uso general como operaciones matemáticas, control de flujo, variables, texto, etc

La mochila sirve para guardar bloques arrastrándolos a ella

Bloques específicos para cada componente en pantalla

Acá nos aparecen avisos de errores de programación en la app
El símbolo de alerta amarillo avisa de bloques que podrían causar un problema
El símbolo de error rojo avisa de errores que impedirán que la app funcione

Se pueden eliminar bloques arrastrándolos al basurero



Con esto terminamos el recorrido por los espacios de trabajo de la plataforma, no tengas miedo de explorarla para conocer todas las funciones que te ofrece.

Curso MIT App inventor 2 - Clase N° 1

Hake Jagurate

En la presente clase vamos a crear una pequeña aplicación que hace que al tocar la foto de un jaguarete, se reproduzca un sonido. Para esta clase necesitarás los archivos "jaguarete.png" y "miau.mp3", si aún no los tienes puedes descargarlos de esta página:

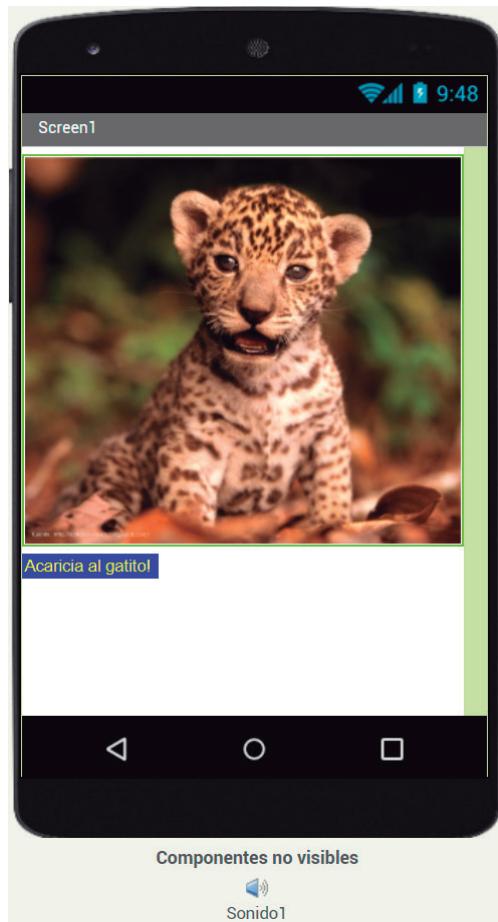
<https://bit.ly/3tAR6uO>

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus propiedades:

Componente	Objetivo	Donde está	Propiedades a editar
 Botón	Mostrar la foto del jaguarete	Interfaz de usuario	Imagen: jaguarete.png Ancho: Ajustar al contenedor Texto: Borrar el texto actual
 Etiqueta	Mostrar el mensaje "Acaricia al jaguarete"	Interfaz de usuario	Color DeFondo: Azul Texto: Acaricia al jaguarete ColorDeTexto: Amarillo
 Sonido	Reproducir el sonido	Medios	Origen: miau.mp

Tu app se va a ver así:



Programando la App

Ahora que tenemos todos los elementos, abrimos la sección *Bloques* con el botón que se encuentra arriba, a la derecha de la página.

Hacé click en "Botón1" en la sección de bloques y verás todos los bloques que hay para el componente de botón.

Vamos a hacer que, cuando al "Botón1" se le de clic, es decir, que se presione con el dedo, se produzca una acción, para eso necesitamos el bloque que detecta un clic en el botón.

La acción que queremos que suceda es que se reproduzca un sonido, para eso necesitamos los bloques que le corresponden al elemento de sonido, entonces haz clic en "Sonido1" para ver sus bloques, busca uno que controle la reproducción del sonido y le dé la orden de reproducirse.

Tus bloques deberían quedar así:



Con esto, la aplicación debería estar funcionando, es momento de probarla.

¡Ahora a divertirte con tu primer juego para celular!

¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarla

1. Cambia el color del texto y su fondo y centra el texto en pantalla para que se vea más linda la aplicación.
2. Hacé que el celular vibre cada vez que se reproduzca el sonido.
3. Cambia la imagen y el sonido del gatito por otras, tal vez un Perrito que ladra, un mosquito que zumba, un bebé que llora, etc. Podés descargar imágenes gratuitas de este sitio web <https://pixabay.com/> y efectos de sonido de estos otros: <https://sfx.freeaudiolibrary.com/> y <https://www.fesliyanstudios.com/>

Curso MIT App inventor 2 - Clase N° 2

Atrapa al Ao Ao

En esta clase vamos a crear un juego en el que tenemos que atrapar al Ao Ao tocando en la pantalla. Para esta clase necesitarás los archivos "Ao_Ao.png", ademas de los sonidos y fondo del juego, si aún no los tienes puedes descargarlos de esta página: <http://bit.ly/3HGR7GM>

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus propiedades:

Componente	Objetivo	Donde está	Propiedades a editar
 Lienzo	Será el área de juego donde aparecerá el aoao	Dibujo y animación.	Alto: ajustar al contenedor. Ancho: ajustar al contenedor. Foto: Cueva.jpg
 SpritelImagen	Los sprites son imágenes que pueden moverse	Dibujo y animación.	Foto: Ao_Ao.png Alto: 90px Ancho: 55px
 DisposiciónHorizontal	Permite ubicar elementos uno a lado del otro.	Disposición	Alto: Automatico Ancho: Automatico
 Botón	Permitirá reiniciar el juego	Interfaz de usuario	Alto Ancho Foto: reset-icon.png
 Etiqueta	Contiene texto que indica su puntuación al jugador.	Interfaz de usuario	Agregar cuatro copias de este elemento dentro de la disposición horizontal. Renombrar como "Aliento" al primero, como "Enegia" al segundo, como "Puntos" al tercero y "Valor_puntos" al cuarto "Energia": Alto: ajustar al contenedor. Ancho: 100px
 Reloj	Con su método temporizador vamos a mover el personaje	Sensores	IntervaloDelTemporizador: 550
 Sonido	Reproducir el sonido	Medios	Crearemos dos, uno de "golpe" y otro de "fin" "Fin" Origen: Grito.mp3 "Golpe" Origen: Golpe.wav

Tu app se va a ver así:



Programando la App

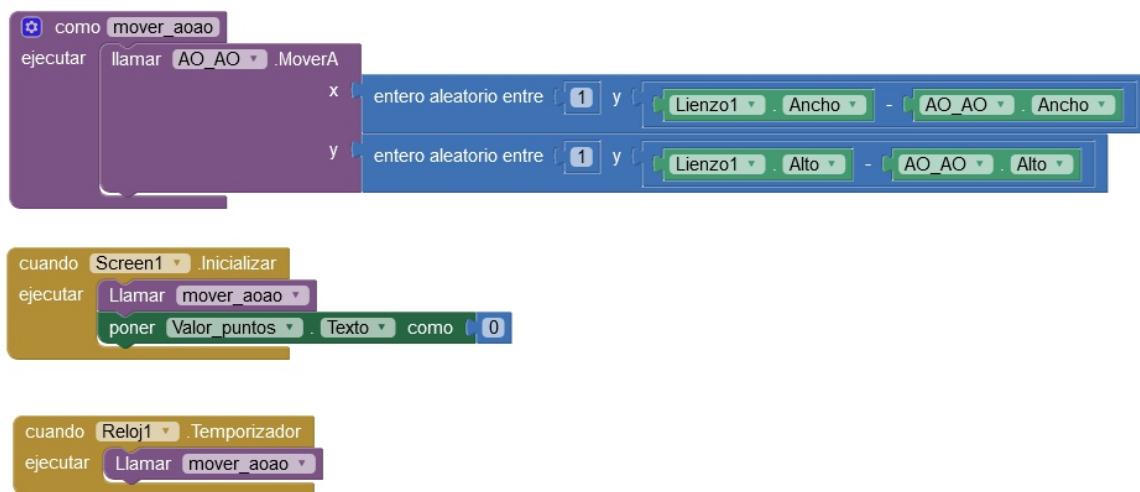
Este es un conjunto de instrucciones que inicializan y manipulan una variable global llamada "energía" y otros elementos de interfaz de usuario.

1. Inicializa la variable global "energía" con un valor de 100.
2. Cuando se haga clic en el botón "Reset", se ejecutará el código que asigna el valor de 100 a la variable "energía" nuevamente.
3. Establece el ancho de un objeto "Energía" para mostrar el valor de la variable "energía".
4. Establece el texto de "Valor_puntos" para mostrar 0
5. Inicializa y pone en marcha "Reloj1"



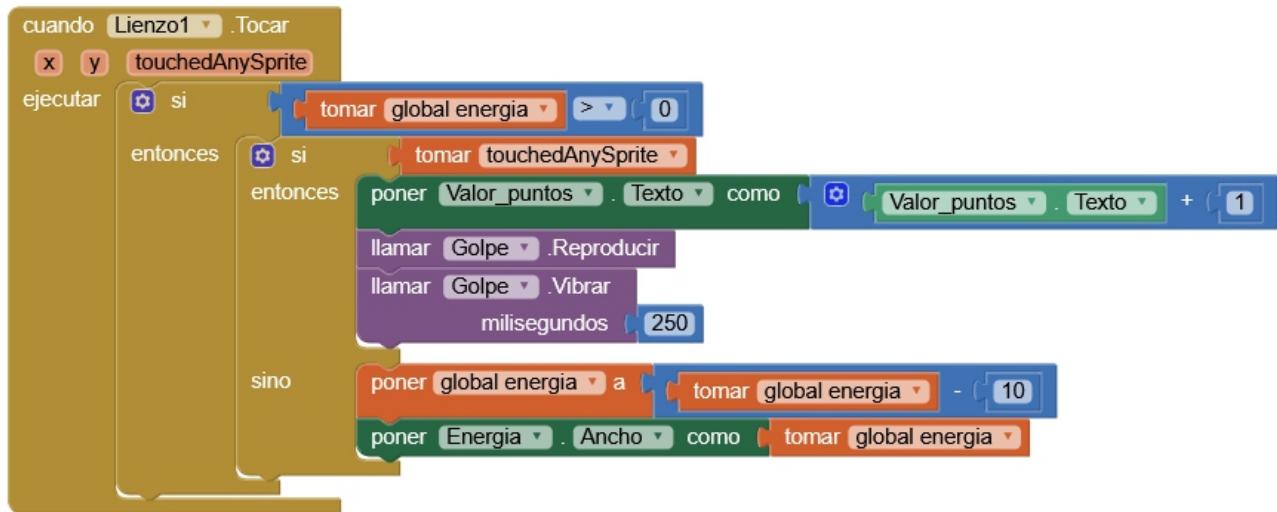
Controla el movimiento del Ao Ao en la pantalla de juego.

1. La función "mover_aoao" se encarga de mover el aoao en una dirección aleatoria.
2. La función "AO_AO" se llama dentro de "mover_aoao" y "Screen1.Inicializar" para mover el aoao en una dirección aleatoria.
3. La función "MoverA" también se llama dentro de "mover_aoao" para mover el aoao en una dirección específica.
4. La variable "AO_AO" se utiliza para almacenar una coordenada aleatoria en el eje X y Y, dentro del rango del ancho y alto del lienzo.
5. Cuando se inicializa la pantalla "Screen1", se ejecuta la función "mover_aoao" para colocar el aoao en una posición inicial aleatoria.
6. La etiqueta "Valor_puntos" se establece en "0" cada vez que se inicializa la pantalla "Screen1".
7. La posición del Ao Ao se actualiza en función del lienzo y sus dimensiones.



Controla lo que sucede cuando se toca el lienzo o al Ao Ao.

Cuando se toca el lienzo "Lienzo1", se ejecuta el código dentro del bloque "si". Se toma el valor de la variable global "energía", si el valor de "energía" es mayor que cero, se verifica si el usuario toco un sprite "AoAo" 1 si es así, se suma 1 a "Valor_puntos" y se reproduce y vibra el efecto de "golpe". De lo contrario si no toca el sprite a el global energía se le restan 10, se ajusta el ancho de la barra de salud al ancho de enegia.



¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarlala

1. Cambia la velocidad a la cual se mueve el Ao Ao
2. Cambia la cantidad de enegia que se pierde al no tocar correctamente
3. Cambia la imagen por la que quieras o añade otro personaje.

Podés descargar imágenes gratuitas de este sitio web <https://pixabay.com/> y efectos de sonido de estos otros:
<https://sfx.freeaudiolibrary.com/> y <https://www.fesliyanstudios.com/>

Curso MIT App inventor 2 - Clase N° 3

Ñamoha'anga

En esta oportunidad vamos a crear una aplicación que nos permite dibujar diferentes formas y colores sobre la pantalla luego vamos a modificarla para que puedas tomar una foto y dibujar sobre ella.

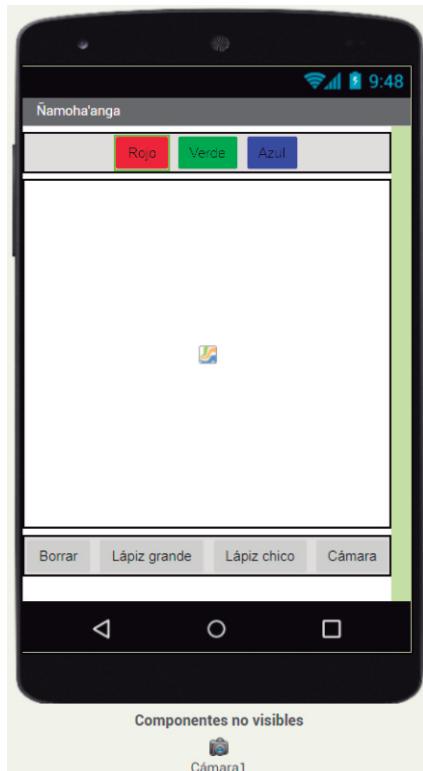
Como app inventor no permite símbolos ni letras en español en los nombres de los proyectos vamos a evitar el pusó (') y nombraremos el proyecto como mohaanga.

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus propiedades:

Componente	Objetivo	Donde está	Propiedades a editar
Screen1	La pantalla contiene los elementos visuales de la aplicación, como botones, imágenes y etiquetas de texto.	Se genera automáticamente al crear el proyecto	Título: Ñamoha'anga
DisposiciónHorizontal	Ubica los elementos uno a lado del otro.	Disposición	Agregar dos copias de este elemento que se llamarán "DisposiciónHorizontal1" que contendrá los botones de colores y "DisposiciónHorizontal2" que contendrá los botones de dibujo. En ambos modificar: DispHorizontal: Centro Ancho: ajustar al contenedor
Botón	Habrá siete botones en la aplicación que nos permitirán elegir el color y tamaño de la línea para dibujar, borrar el dibujo que hicimos o tomar una foto.	Interfaz de usuario	Agregar 3 botones en "DisposiciónHorizontal1": ColorDeFondo: cambiar cada uno a rojo, verde y azul Texto: modificar según el color de fondo de cada botón Ancho: ajustar al contenedor Agregar 4 botones en "DisposiciónHorizontal2": Texto: Borrar, Lápiz grande, Lápiz chico y cámara. Renombrar cada botón según su propiedad Texto.
Lienzo	Es el área de la pantalla donde podemos dibujar.	Dibujo y animación	Ubicar este elemento en el medio de los dos componentes de disposición que se agregaron anteriormente. Ancho: ajustar al contenedor Alto: 300px
Cámara	Inicia la cámara de tu teléfono para permitirte tomar una foto.	Medios	No se modifica.

Tu app se va a ver así:



Programando la App

Una vez terminado el diseño de la app nos vamos a los bloques.

Primeramente vamos a hacer que el lienzo tome el color de cada botón para dibujar, para eso, cuando demos clic en cada botón le asignamos su color correspondiente a la propiedad *ColorDePintura* del lienzo, los colores los vas a encontrar en la sección Colores de los bloques. Debe quedar así:



Seguidamente, programamos los botones de borrar, lápices y cámara.

Nuevamente necesitamos programar qué sucede al darle clic a cada botón por lo que necesitaremos los bloques del método *Clic* de cada uno, para el botón de borrar, le indicaremos al lienzo que limpie lo que hemos dibujado con su método *Limpiar* y para el botón cámara, le indicaremos al componente cámara que tome una foto mediante su método *TomarFoto*, así:



Una vez tomada la foto, debemos decirle al lienzo que la use de fondo para poder dibujar sobre ella, para eso necesitamos el bloque del método *DespuésDeTomarFoto* de la cámara, que le indica qué hacer con ella, en este caso, la pondremos como fondo del lienzo para dibujar sobre ella mediante el bloque de método *poner ImagenDeFondo como* del lienzo.

Este método recibe un valor para funcionar, a esto se le llama parámetro, en este caso nuestro parámetro es la foto que fue tomada, para obtenerla, debemos dejar el cursor sobre la palabra *imagen* que está en el bloque, veremos que se despliega una ventanita con dos bloques, usaremos el que dice *tomar imagen* y lo unimos al método del lienzo mencionado anteriormente.



Lo siguiente es programar los botones para controlar el tamaño del lápiz, para esto primero necesitamos a crear una variable, las variables sirven para almacenar información que será usada por el programa, esta información se guarda en la memoria de tu celular y se queda ahí mientras la aplicación esté funcionando.

Para crear una variable vamos a ir al apartado de Variables de los bloques, arrastramos el bloque *Iniciar global nombre como* y modificaremos “nombre” cambiándolo a “tamanho_lapiz”, el siguiente paso es conectarle un bloque de número que se encuentra en el apartado de matemática, este bloque lo modificamos dándole el valor 2 y de esta manera creamos nuestra variable llamada “tamanho_lapiz” con el valor 2. Te debe quedar así:

iniciar global tamanho_lapiz como 2

Ahora que tenemos lista nuestra variables podemos cambiar el tamaño del lápiz a nuestro gusto al presionar los botones “Lápiz grande” o “Lápiz chico”, estos botones deben modificar el valor de la variable “tamanho_lapiz” asignándole un valor más grande o más pequeño según qué botón se presione.

Para eso vamos a buscar los bloques que verifican que se haya dado clic en el botón y les indicaremos asignarle un valor de 2 u 8 a nuestra variable, el bloque para cambiar el valor de la variable se encuentra en la sección Variables de los bloques.

Al finalizar tendrás esto:

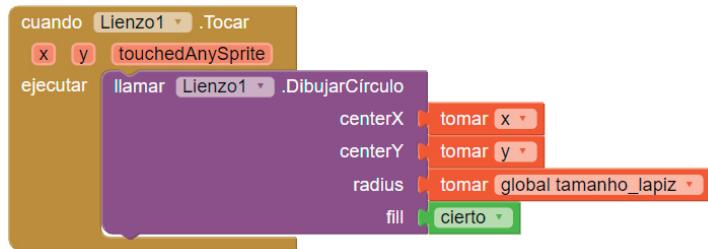


Enseñandole a la app como dibujar:

Ya casi terminamos, solo nos falta mostrarle a la app cómo dibujar, porque eso es lo que hacemos cuando programamos, enseñarle a la computadora o el celular qué debe hacer y cómo hacerlo.

Entonces, ¿cómo se dibuja?.

Cuando dibujamos hacemos líneas y puntos, primero vamos a decirle al lienzo cómo debe dibujar un punto, lo que debe hacer es detectar en qué coordenadas de la pantalla se hizo clic, para eso el lienzo tiene un método llamado *Tocar* que detecta que hemos hecho un toque en la pantalla, y obtiene las coordenadas en x e y del lugar donde tocamos, con esto, llamamos a la función *DibujarCírculo* para que dibuje un punto en esa coordenada, también necesitaremos el tamaño del radio, que será el tamaño de nuestro lápiz, para eso debemos tomar el valor de la variable “tamaño_lapiz” mediante el método *tomar* dentro de los bloques de Variables, por último, al parámetro *fill* (relleno) le asignamos el valor “Cierto” que se encuentra entre los bloques de lógica, para que dibuje un círculo relleno.

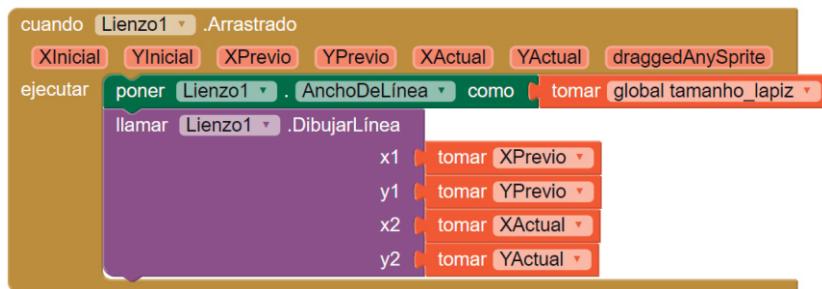


¿Y cómo hacemos las líneas?

Primero vamos a establecer el ancho del trazo modificando la propiedad *AnchoDeLínea* del lienzo, vamos a establecer el ancho con el valor que tenga la variable para que cada al cambiar este valor con los botones de lápiz grande y lápiz chico el trazo varíe de grosor.

Lo siguiente es hacer que se dibujen las líneas y para eso veremos un poco de geomtría, una línea recta o curva se forma con una sucesión de segmentos dibujados uno tras otro luego de arrastrar el dedo por la pantalla una cierta distancia, cuando hacemos esto, se generan dos puntos (coordenadas en x e y), uno de inicio (*XPrevio* e *YPrevio*, que es donde empezamos a tocar la pantalla) y otro de fin (*XActual* e *YActual*, donde soltamos el dedo), el evento *Arrastrado* del lienzo obtiene estas coordenadas y con ellas traza pequeñas líneas que unidas forman el trazo que se hace arrastrando el dedo, así podremos dibujar una línea mediante el método *DibujarLínea* del lienzo.

Debe quedar de esta forma:



Con esto, la aplicación debería estar funcionando, es momento de probarla.

¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarlala

1. Crea una etiqueta para mostrar el valor actual del tamaño del lápiz
2. Agrega una etiqueta que indique el color actual del lápiz
3. Permite al usuario agregar un tamaño de lápiz personalizado así el lápiz podrá variar de tamaño según el valor introducido y no solo entre 2 y 8.

Curso MIT App inventor 2 - Clase N° 4

Echo Coronavirus!

Hoy haremos una nueva versión de un juego que tal vez ya conocés, ¿alguna vez escuchaste del juego de golpear al topo?, es un juego en el que irán apareciendo unos topos en pantalla y debes tocarlos para sumar puntos, pero nosotros en vez de golpear topos vamos a golpear al coronavirus por habernos puesto en cuarentena tanto tiempo.

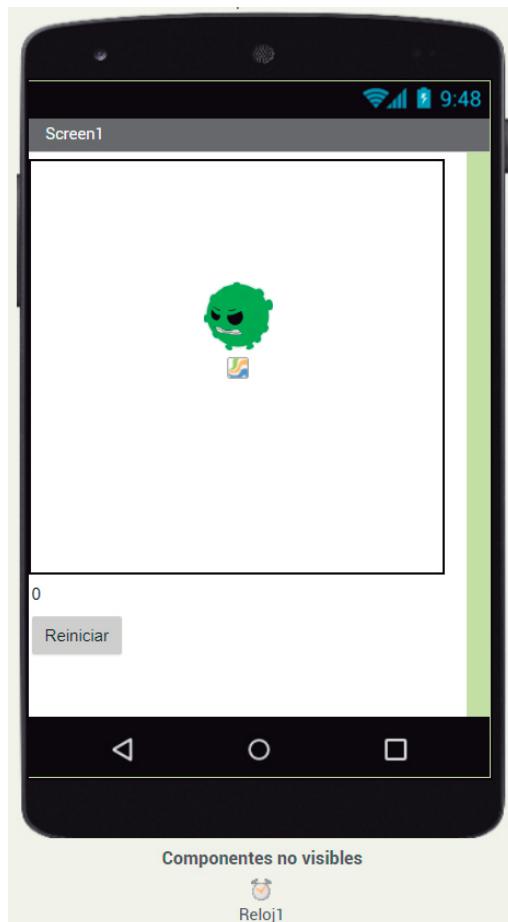
Para esta clase vas a necesitar la imagen "covid1.png" si aún no la tienes puedes descargarla de acá: <https://bit.ly/3tAR6uO>

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus propiedades:

Componente	Objetivo	Donde está	Propiedades a editar
 Lienzo	Es el área de la pantalla donde va a aparecer el coronavirus.	Dibujo y animación	Ancho: ajustar al contenedor Alto: 300px
 SpritelImagen	Es una imagen que puede moverse en pantalla, debe arrastrarse dentro del lienzo.	Dibujo y animación	Foto: covid1.png Ancho: 50 px. Alto: 50 px.
 Botón	Permitirá reiniciar el juego poniendo el contador de la puntuación en 0.	Interfaz de usuario	Renombrar a Reiniciar Texto: Reiniciar
 Etiqueta	Contiene texto que muestra el puntaje del jugador.	Interfaz de usuario	Texto: 0
 Reloj	Usaremos el temporizador del reloj para mover al virus de lugar.	Sensores.	IntervaloDelTemporizador: 500

Tu app se va a ver así:



Programando la App

Una vez terminado el diseño de la app nos vamos a los bloques.

Para empezar a programar nuestro juego debemos definir el comportamiento del sprite del virus, queremos que aparezca en pantalla de forma aleatoria cada cierto tiempo, para eso vamos a crear un procedimiento que mueva al elemento a una coordenada generada al azar.

Para lograrlo vamos a crear un procedimiento donde vamos a indicarle al sprite que se mueva a una coordenada x e y del lienzo, como estas coordenadas deben ser generadas al azar utilizaremos el bloque *entero aleatorio entre* con los valores 0 y *tamaño del lienzo - tamaño del sprite* (el tamaño del sprite lo obtenemos de los bloques ancho o alto del mismo según sea la coordenada en x o y), hacemos este cálculo para que el sprite no se salga de pantalla.

Estos bloques que generan los números los adjuntamos a los métodos *poner x como* y *poner y como* del sprite.

Finalmente, arrastramos el bloque *como procedimiento ejecutar* de las sección Procedimientos de los bloques y metemos los bloques anteriores dentro de este, cambia el nombre procedimiento por MoverVirus y listo. Tiene que quedar así:

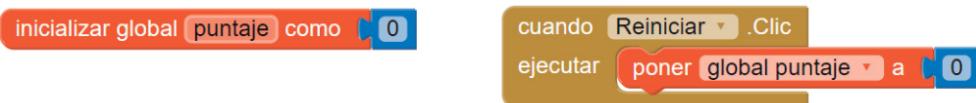


Los procedimientos son útiles para realizar acciones repetitivas sin tener que crear los mismos bloques de código una y otra vez, simplemente utilizamos el procedimiento creado haciendo lo que se denomina una llamada a un procedimiento.

Para llamar al procedimiento usaremos el bloque *llamar <nombre del procedimiento>* disponible en la sección Procedimientos de los bloques, pero esto le haremos más adelante.

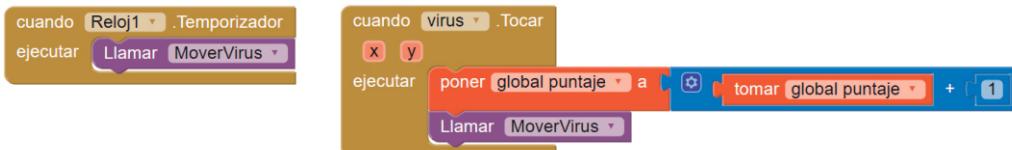
El siguiente paso es crear una variable que va a contener el puntaje, para eso vamos a la sección de Variables, elegimos el bloque *inicializar global <nombre> como* y nombramos a nuestra variable puntaje, esta variable nos ayudará a llevar la cuenta de nuestra puntuación incrementándose en 1 cada vez que golpeamos al virus.

Así también, debemos ponerla nuevamente en 0 cuando presionemos el botón Reiniciar.



Ya tenemos las coordenadas y el puntaje, es momento de trabajar en el comportamiento del sprite, lo que necesitamos es que cada vez que pase un determinado tiempo el virus se mueva de lugar, vamos a usar el método *Temporizador* del reloj para esto, anteriormente definimos su intervalo en 500 milisegundos, esto significa que este método ejecutará una acción pasado ese tiempo, es acá donde llamamos a nuestro procedimiento que mueve al virus.

Por otro lado, cuando toquemos al sprite del virus debemos sumar 1 al puntaje y mover de nuevo al virus.



Solo nos falta un detalle, debemos tener una forma de reflejar el nuevo puntaje cada vez que cambie, para eso vamos a crear otro procedimiento llamado ActualizarPuntaje que va a modificar el texto de la etiqueta de puntaje cuando se llamado.



Por último, llamamos a este procedimiento donde sea necesario, en este caso sería al reiniciar, porque debemos actualizar la etiqueta para que diga 0 de nuevo y también cada vez que sumemos un punto, en la siguiente hoja verás donde debes agregar estas llamadas.



Con esto, la aplicación debería estar funcionando, es momento de probarla.

¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarla

1. Cambia la velocidad de aparición del virus.
2. Agrega un efecto de sonido cuando el virus es golpeado, podés descargar efectos de sonido gratuitos de estos sitios:
<https://sfx.freeaudiolibrary.com/> y <https://www.fesliyanstudios.com/>
3. Cuando golpees al virus, cambia la imagen al virus muerto llamada "covid2.png", debes hacer que espere unos segundos antes de moverse para poder ver el cambio, vas a encontrar la imagen acá <https://bit.ly/3tAR6uO>

Curso MIT App inventor 2 - Clase N° 5

Plata yvyguy

En esta clase vamos a crear un juego en el que un buscador de plata yvyguy tiene que cavar para recolectar todo el oro en la pantalla. Para esta app vas a necesitar los archivos "heka.png" y "viru.png", si aún no los tienes puedes descargarlos de esta página:

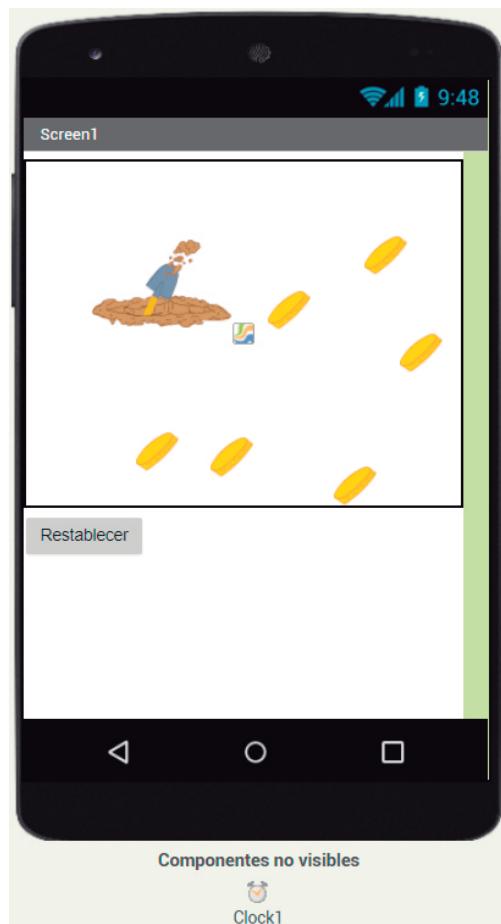
<https://bit.ly/3tAR6uO>

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus propiedades:

Componente	Objetivo	Donde está	Propiedades a editar
 Lienzo	Será el área de juego donde aparecerá el tesoro.	Dibujo y animación.	Alto: 350 px. Ancho: ajustar al contenedor.
 SpritelImagen	Los sprites son imágenes que pueden moverse, serán el personaje y el tesoro del juego.	Dibujo y animación.	Colocar 7 copias de este elemento en pantalla Foto: viru.png a 6 de ellos y heka.png al elemento restante Renombrar cada moneda como viru1, viru2, viru3, etc. y como heka al sprite de la pala Velocidad: cambiar la velocidad de heka a 6
 Reloj	Usaremos el reloj como temporizador para mover las monedas.	Sensores.	No se modifica
 Botón	Reiniciará el juego para que podamos volver a jugar una vez capturado todo el tesoro.	Interfaz de usuario.	Renombrar a Restablecer Texto: Restablecer

Tu app se va a ver así:



Programando la App

Una vez terminado el diseño de la app nos vamos a los bloques.

En este juego somos buscadores y buscadoras de un tesoro que se encuentra enterrado en alguna parte de la pantalla según cuenta la leyenda de plata yyguy, ¿te acordás de ella?, para cavar debemos movernos a través de la pantalla arrastrando nuestra pala en dirección al oro que va apareciendo en distintos lugares, debemos recolectar todo el oro para ganar.

Para movernos vamos a “lanzar” el elemento heka en la dirección que queramos, para eso necesitamos el bloque controlador de eventos *Lanzado* que detecta que el elemento está siendo arrastrado en una dirección.

Este bloque toma 6 atributos: x, y, velocidad, dirección, velocidadx y velocidady que son valores obtenidos al arrastrar el elemento heka. Queremos reasignar la dirección actual de heka al rumbo que nos dio el bloque *Lanzado*, esto significa que ahora se puede controlar la dirección de la pala con los dedos al lanzar la pantalla.



Y para evitar que la excavación se salga de la pantalla usaremos el bloque *Botar* cuando se alcance un borde.



También queremos que las monedas se muevan a posiciones aleatorias en la pantalla. Utilizaremos el método *Temporizador* del reloj y el método *MoverA* de SpritelImagen hacer esto.

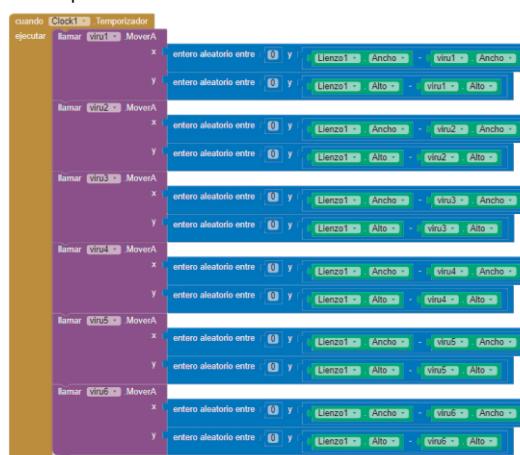
MoverA toma dos argumentos: las coordenadas x e y en el lienzo de la nueva posición a la que queremos que se mueva el componente SpritelImagen, esto es como ubicar un punto en el plano cartesiano y las coordenadas son puntos en la pantalla, así:



Cuando se apaga el temporizador, queremos que todas nuestras monedas de oro se muevan a una nueva ubicación aleatoria en el lienzo con el método *MoverA* de esta forma:



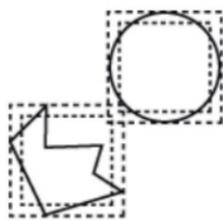
Repetimos estos bloques para cada moneda quedando así:



Comportamiento del juego:

El juego funciona detectando colisiones, una colisión es cuando dos elementos se tocan entre sí. App inventor detecta colisiones al verificar una intersección entre los rectángulos delimitadores de cada SpritelImagen. A esto lo llamamos detección de colisión basada en rectángulos y funciona de esta forma:

No hay colisión



Colisión detectada

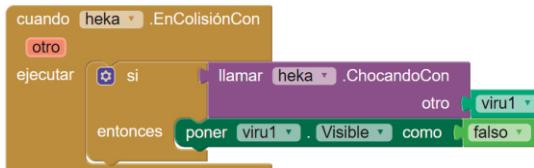


Podemos usar el controlador de eventos *EnColisiónCon* para detectar cada vez que nuestra excavación colisiona con una moneda de oro. Fíjate que *heka.ChocandoCon* toma un parámetro, este será el elemento con el que acaba de chocar, en este caso, el SpritelImagen de alguna de las monedas de oro.

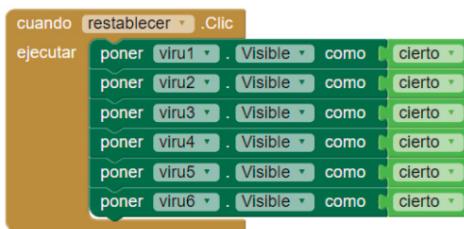
Cada vez que el pirata choco con una moneda de oro, queremos que la moneda desaparezca. Podemos hacer esto configurando la visibilidad de la moneda como "falso".

Para encontrar con qué moneda chocó nuestra pala, usaremos el método *heka.ChocandoCon* y le preguntaremos si ha chocado con cada uno de los sprites de monedas de oro para detectar qué sprite fue golpeado. Para hacer esto usaremos un bloque de componente que hace referencia a un sprite. Este bloque se puede encontrar en el cajón de bloques de cada componente.

Si se golpeó una moneda, estableceremos su visibilidad en falso (como se ve en la primera imagen) y repetiremos esto para cada moneda, quedando como en la segunda imagen:



Por último, configuramos un botón de reinicio que hará nuevamente visibles a todas las monedas, para eso, al hacer clic en el botón cambiamos la visibilidad de todas las monedas a verdadero:



Con esto, la aplicación debería estar funcionando, es momento de probarla.

¡Ahora a divertirte con tu nuevo juego para celular!

¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarla

1. Agrega un texto para mostrar el tiempo que te llevó obtener todo el oro
2. Agrega una piedra que, al golpearla, hace que caves más lento por unos segundos.
3. Usa uno de los sensores del teléfono para controlar el movimiento de la pala.

Curso MIT App inventor 2

Space Invaders

Esta vez vamos a crear un juego inspirado en uno de los videojuegos más famosos del mundo, Space invaders, su título significa inasores espaciales, es un juego de 1978 donde debes dispararles a unas naves espaciales alienígenas.

Para esta clase vas a necesitar las imágenes "nave.png" y "cohete.png", si aún no la tienes puedes descargarla de acá:

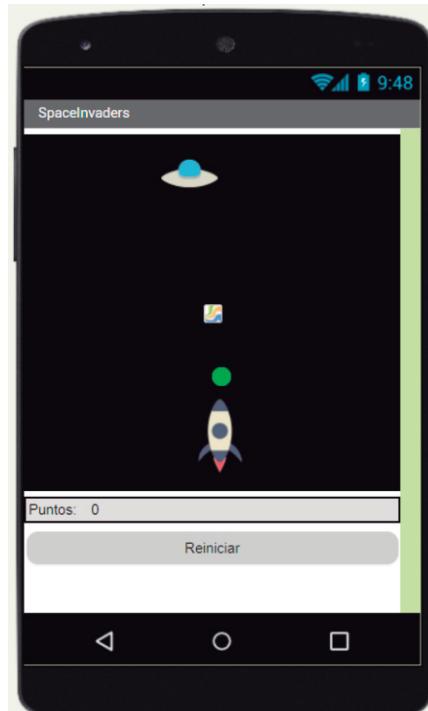
<https://bit.ly/3tAR6uO>

Creando la pantalla y componentes:

Vamos a arrastrar a la pantalla los siguientes componentes y a editar sus pr

Componente	Objetivo	Donde está	Propiedades a editar
Lienzo	Es el área de la pantalla donde podemos jugar, va a contener a los sprites y la pelota que hará de bala.	Dibujo y animación	ColorDeFondo: Negro Ancho: ajustar al contenedor Alto: 300px
SpritelImagen	Es una imagen que puede moverse en pantalla, serán las naves.	Dibujo y animación	Agrega dos copias de este elemento Renombrar uno de ellos como nave y el otro como cohete Foto: nave.png y cohete.png respectivamente
Pelota	Es un sprite de una esfera, será la bala del cohete.	Dibujo y animación	ColorDePintura: Verde Radio: 8 Velocidad: 10
Botón	Permitirá reiniciar el juego poniendo el contador de la puntuación en 0.	Interfaz de usuario	Renombrar a Reiniciar. Ancho: Ajustar al contenedor Texto: Reiniciar.
DisposiciónHorizontal	Permite ubicar elementos uno a lado del otro.	Disposición	Ancho: ajustar al contenedor
Etiqueta	Contiene texto que indica su puntuación al jugador.	Interfaz de usuario	Agregar dos copias de este elemento dentro de la disposición horizontal. Renombrar como "texto_puntaje" al primero y como "valor_puntaje" al segundo Texto: para "texto_puntaje" escribir Puntos y para "valor_puntaje" escribir 0
Reloj	Con su método temporizador vamos a mover la nave espacial.	Sensores	IntervaloDelTemporizador: 3000

Tu app se va a ver así:



Programando la App

Una vez terminado el diseño de la app nos vamos a los bloques.

El comportamiento de la app se divide en tres partes: el cohete, la nave y la pelota que hará de bala.

Primero programaremos el comportamiento del cohete. Queremos que el cohete pueda moverse de izquierda a derecha al ser arrastrado, para eso vamos a usar el bloque *Arrastrado* que va a tomar la coordenada en x del punto en el que hayamos soltado el sprite y moverlo a esa coordenada mediante el método *poner x como* del sprite.

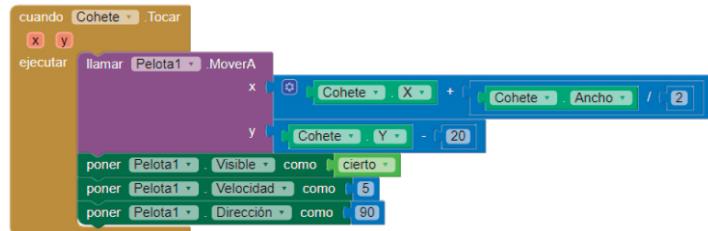


Ahora vamos a programar el comportamiento de la bala.

Queremos que la bala sea invisible al iniciar el juego, justo antes de disparar por primera vez y también que desaparezca cuando golpee la nave para dar el efecto de que impactó en ella, para el primer caso utilizaremos el bloque *Iniciarizar* de la pantalla, cuando ésta se inicie, queremos que la pelota tenga su propiedad *Visible* como *falso*. También queremos que esto pase si la pelota llega al otro lado de la pantalla sin colisionar con la nave, es decir, tocó un borde:



A continuación vamos a programar lo que pasa cuando tocamos el cohete: la bala debe salir del cohete al tocarlo, cuando salga debe tener una velocidad y dirección que estableceremos de la siguiente manera:

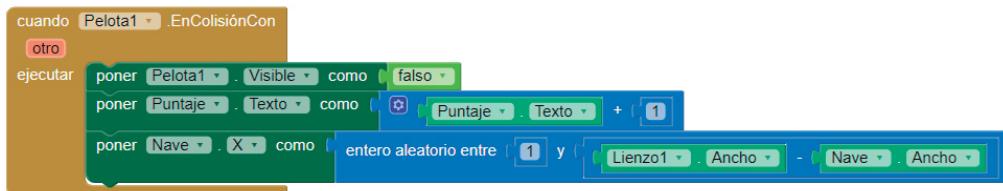


Donde *MoverA* mueve la pelota desde el medio de la posición del cohete, esto lo calcula tomando su coordenada en x desde la esquina superior izquierda y sumándole la mitad del ancho del sprite. En el eje y, calcula la altura del sprite menos 20 píxeles, para que el movimiento de la bala se vea más realista apareciendo desde debajo de la nave.

Luego hacemos visible a la pelota, le damos una velocidad de movimiento de cinco píxeles por vez y le asignamos una dirección, los valores de dirección están basados en el plano cartesiano y van de 0 a 360 donde 0 es el oeste, 90 es el norte, 180 es el este y 270 es el sur, en este caso, como queremos que vaya hacia arriba (norte), asignamos el valor 90.

Lo siguiente es detectar que la bala ha impactado la nave, esto lo hacemos mediante colisiones.

La pelota tiene un bloque que detecta que ha colisionado con otro sprite, entonces cuando este evento suceda simplemente debemos hacer invisible a la pelota, sumar 1 al puntaje y moverlo de nuevo a la posición inicial detrás del cohete:



¡Ya casi terminamos!, solo nos falta mover a la nave y programar el botón reiniciar, son apenas cuatro instrucciones restantes que están en la siguiente página.

Para mover la nave usaremos el temporizador, cada vez que se acabe el tiempo, fijaremos la coordenada en x de la nave a un punto aleatorio entre 1 y el ancho del lienzo - el ancho del sprite (para que no se nos salga de la pantalla).



Y haremos que el botón reiniciar ponga el puntaje en 0 nuevamente:



Con esto, la aplicación debería estar funcionando, es momento de probarla.

¡Desafíos!

Ahora que ya terminaste tu aplicación podés mejorarla

1. Cambia la velocidad con la que se mueven la bala y la nave espacial para que el juego sea más rápido.
2. Agrega efectos de sonido cuando el cohete dispara y la nave es golpeada, podés descargar efectos de sonido gratuitos de estos sitios: <https://sfx.freeaudiolibrary.com/> y <https://www.fesliyanstudios.com/>
3. Modifica la apariencia del juego para personalizarlo a tu gusto.