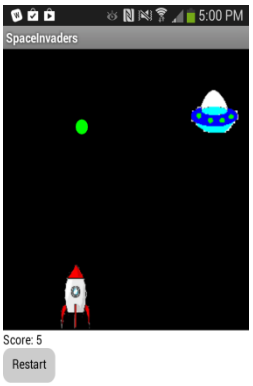
PRÁCTICA 7. VIDEOJUEGO. SPACE INVADERS

Primera parte.

Lo que vas a construir



Con la construcción de la App Space Invaders obtendrás práctica con el uso de componentes como el reloj y temporizadores, utilizando componentes de animación tales como imágenes Sprites y Canvas, el establecimiento de la visibilidad, y la detección de colisiones en App Inventor.

Vas a programar una aplicación que tiene un cohete que dispara a todos los platillos voladores que aparecen en pantalla.

Primeros pasos

Concéctate al sitio Web **App Inventor** y empieza un nuevo proyecto. Define como **SpaceInvaders** , y también establece el título de la pantalla a **"SpaceInvaders"**. Cambia a la vista **Blocks** y conéctate al **emulador o a un dispositivo** .

Introducción

Esta práctica presenta las siguientes habilidades, útiles para el futuro desarrollo del juego:

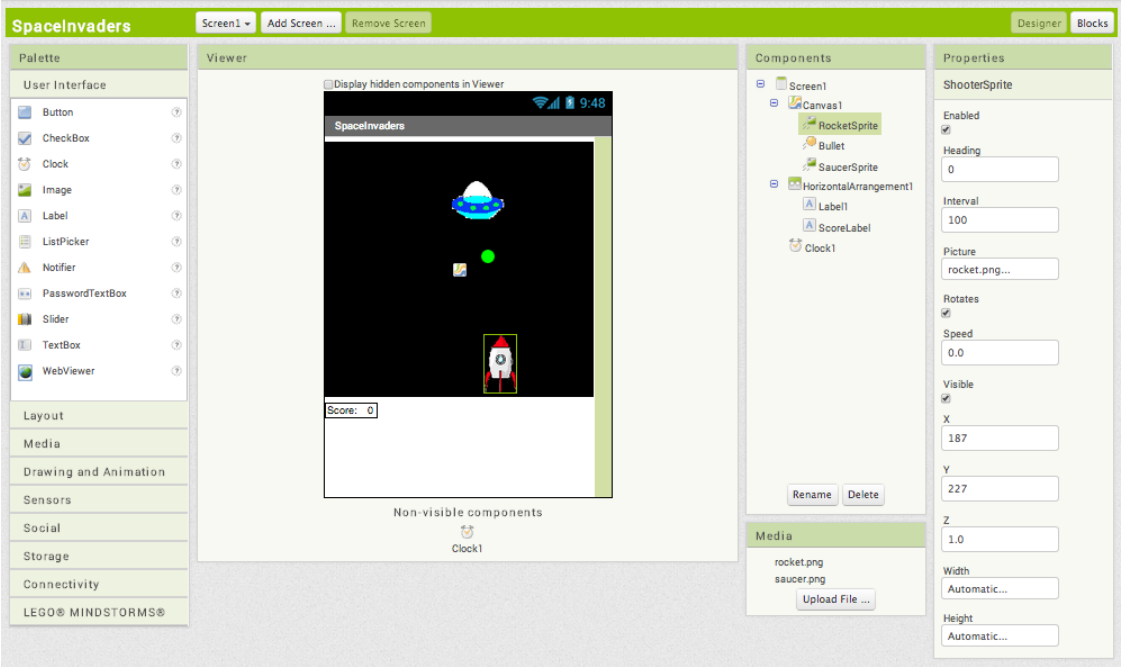
- Utilización del componente **Reloj (clock)** .
- Uso de **Clock.Timer** (temporizadores) para mover sprites.
- Uso de **Sprite.Dragged** para mover un sprite.
- Uso de la **detección de colisiones**.
- Configuración de la **visibilidad** de sprites.

Preparándose

Para este juego, debes tener dos tipos de sprites: una **ImageSprite** representando a un **cohete** que dispara y otra a **platillos** voladores.

Configurar los componentes

Utiliza el Designer para crear la interfaz de **SpaceInvaders** . Cuando esté terminada, debería ser como la imagen de abajo.



Para crear esta **interfaz**, pon los siguientes componentes en el **Designer** arrastrándolos desde la paleta de componentes al la pantalla del **Visor**.

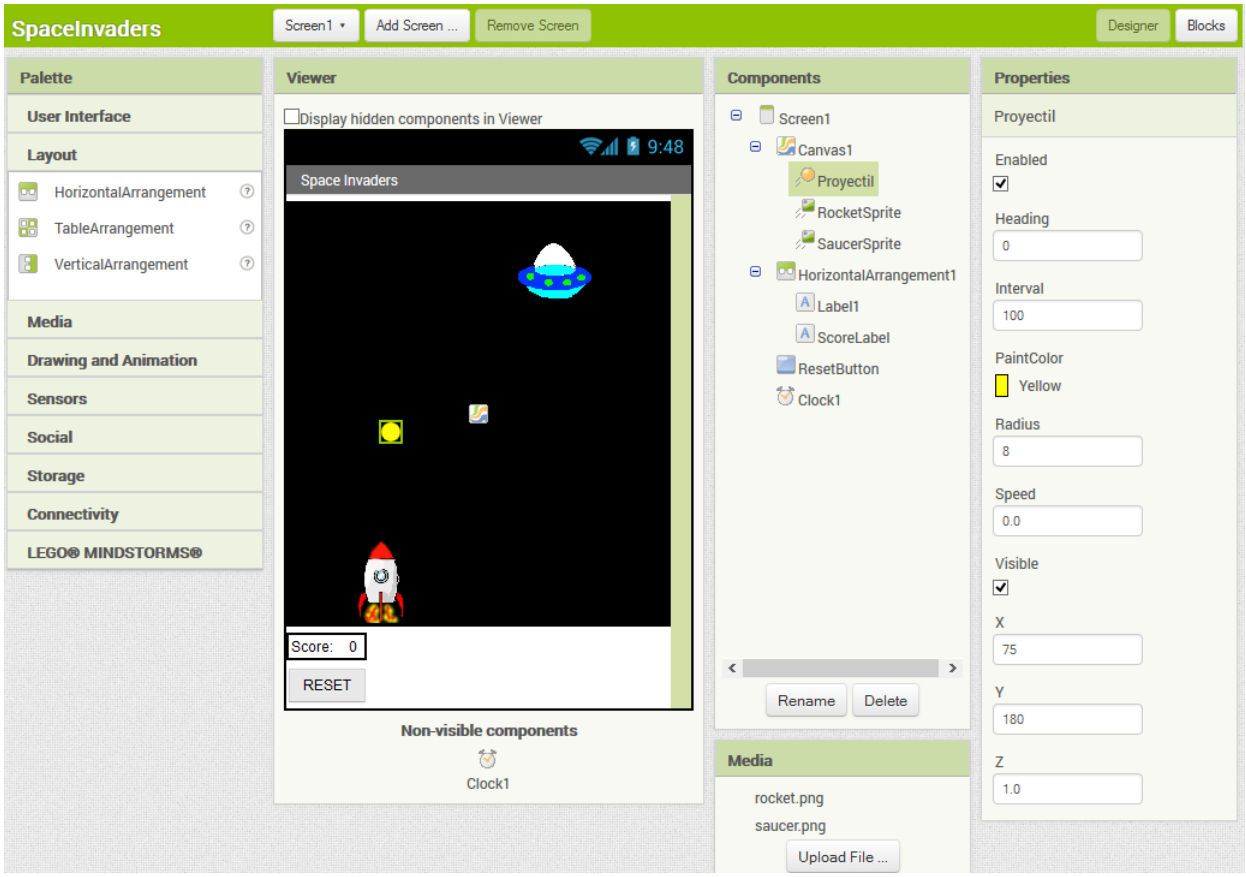
Tipo de componente	Paleta	Nombre	Función que desarrolla
Canvas	Drawing and Animation	**Canvas1**	El fondo que vamos a poner nuestros a nuestro espacio
ImageSprite	Drawing and Animation	**RocketSprite**	La nave espacial en nuestro juego
ImageSprite	Drawing and Animation	**SaucerSprite**	El platillo volador en nuestro juego
Ball	Drawing and Animation	**Proyectil**	El proyectil de la nave espacial.
Clock	User Interface	**Clock1**	Usamos el reloj como temporizador para mover el platillo
HorizontalArrangement	Layout	**HorizontalArrangement1**	Para contener la puntuación Label
Label	User Interface	**Label1**	Para contener la palabra "Score"
Label	User Interface	**ScoreLabel**	Para contener la puntuación numérica actual
Button	User Interface	**ResetButton**	Para reiniciar el juego para que el

jugador puede jugar otra vez

Establece las propiedades de los componentes que se describen a continuación:

Componente	Acción
Canvas1	Cambia el Ancho a "Fill parent". Cambia la Altura a 350 y define el fondo en negro
ResetButton	Cambia el texto a "Reset".
RocketSprite	Cargar la imagen del cohete y establécela a "rocket.png". Ajusta la coordenada Y a 275. Esto hará que el cohete se encuentre en la parte inferior del lienzo.
SaucerSprite	Cargar la imagen del platillo y establécela a "saucer.png".
Ball	Establece el radio a 8, una velocidad inicial de 10 y coloca el color amarillo.
Label1	Cambia el texto a "Score".
ScoreLabel	Cambia el texto a "0".
Clock	Cambia la propiedad TimerInterval a 3000.

Ahora que has configurado todas las propiedades esenciales, no dudes en cambiar los colores de los componentes que desees.



Vamos a mover el cohete

En este juego, el usuario moverá el **cohete** de lado a lado. Esto significa que sólo vamos a cambiar la **coordeanada X** del sprite del cohete. Para ello vamos a utilizar el **when RocketSprite.Dragged**. Cuando se arrastra el cohete, ajustaremos la **coordenada X** al **CurrentX** que detecta el dedo en la pantalla.

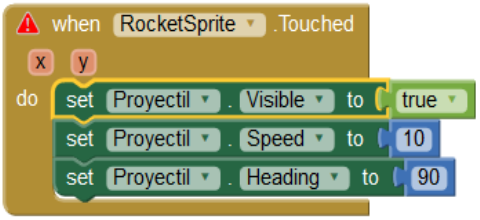
Comprueba el funcionamiento del cohete.

Programación del comportamiento del proyectil

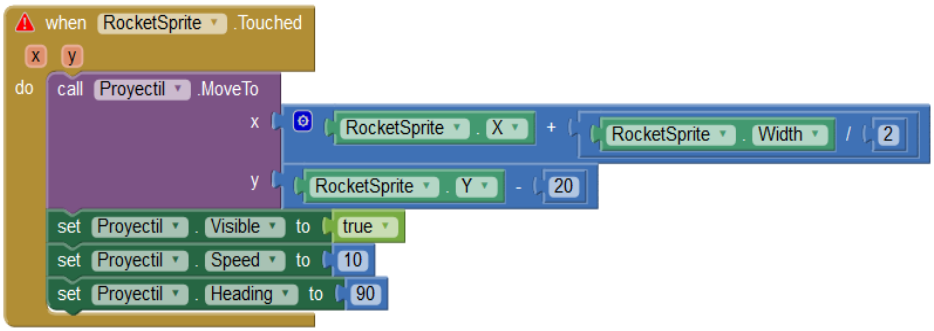
Hay varias características que queremos que nuestros **proyectiles** tengan en este juego. Queremos que se **disparen** desde el **cohete**, **choquen** con el **platillo**, y sean **invisibles**después de la colisión.

Vamos a empezar con el bloque **when Screen1.Initialize** . Cuando se inicializa la pantalla, vamos a programar que el **proyectil** sea **invisible**. Haremos esto mediante el establecimiento de la **propiedad de visibilidad** del proyectil a **" false"**.

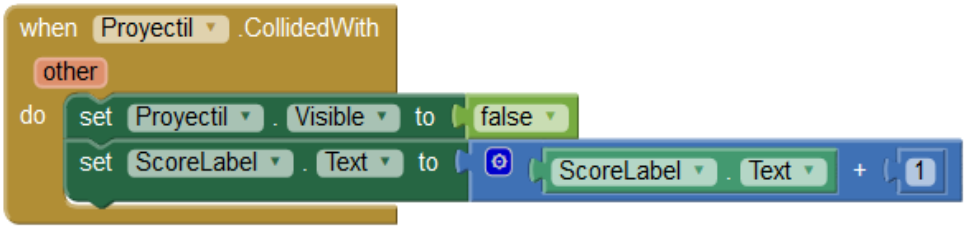
A continuación, queremos asegurarnos de que el **proyectil** vuelve a **aparecer** cuando disparamos desde el **cohete**. Cuando tocamos el cohete, queremos que el **proyectil se dispare hacia el platillo**, lo haremos mediante el uso del controlador **when RocketSprite.Touched.** Cuando se toca el cohete, no sólo queremos establecer que sea **visible**, también queremos establecer la **velocidad y el rumbo del proyectil**. El **águlo** puede ser un valor entre 0 y 360 que indica en qué dirección debería moverse. **90** es hacia arriba. La velocidad se mide en píxeles / seg.



Una vez que se **dispara el proyectil**, no puede disparar de nuevo. Tenemos que programar el proyectil para **volver al cohete** cuando disparamos. Podemos hacer esto usando un bloque de llamada **call Proyectil.MoveTo** .

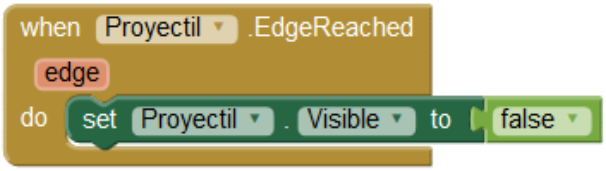


La última cosa que necesitamos programar es lo que sucede cuando el **proyectil choca con el platillo**. Vamos a utilizar el controlador **when Proyectil.CollidedWith**. Este evento se produce cuando el **proyectil choca con otro sprite**. Dado que nuestro sprite de **cohete** está bloqueado en una **coordenada Y** de la parte inferior de la pantalla, el proyectil nunca chocará con el cohete y sólo con el **platillo**. En la **colisión** queremos que sucedan dos cosas, primero que la **puntuación** aumente en 1 y segundo que el proyectil se vuelva **invisible**.



Comprueba su funcionamiento.

Te habrás dado cuenta que cuando **toca el platillo**, **el proyectil** se mueve a la parte superior de la pantalla y se queda **atascado** en ella hasta que se **dispara** de nuevo. Para hacer que el **proyectil desaparezca** cuando llega al **borde superior** de nuestro **Canvas**, tenemos que utilizar el controlador **when Proyectil.EdgeReached.**

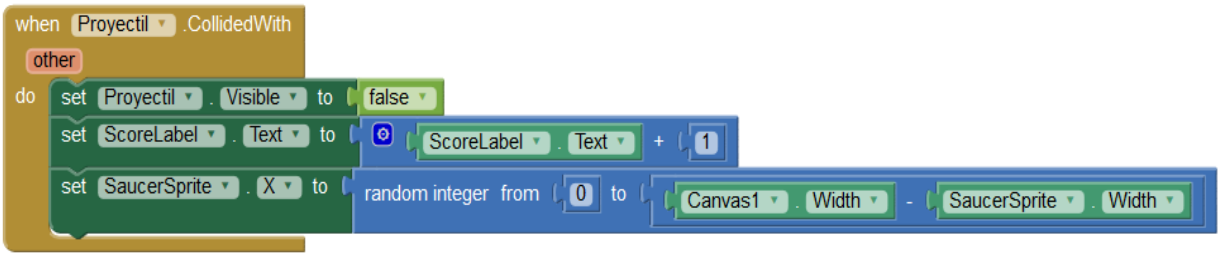


<u>Programación del botón Reset</u>

A veces, los usuarios podrían querer **reiniciar el juego** y reiniciar su puntuación. Cuando esto suceda, es necesario establecer la puntuación de nuevo a 0.

<u>Aumento de la dificultad. Cambio de la posición del platillo</u>

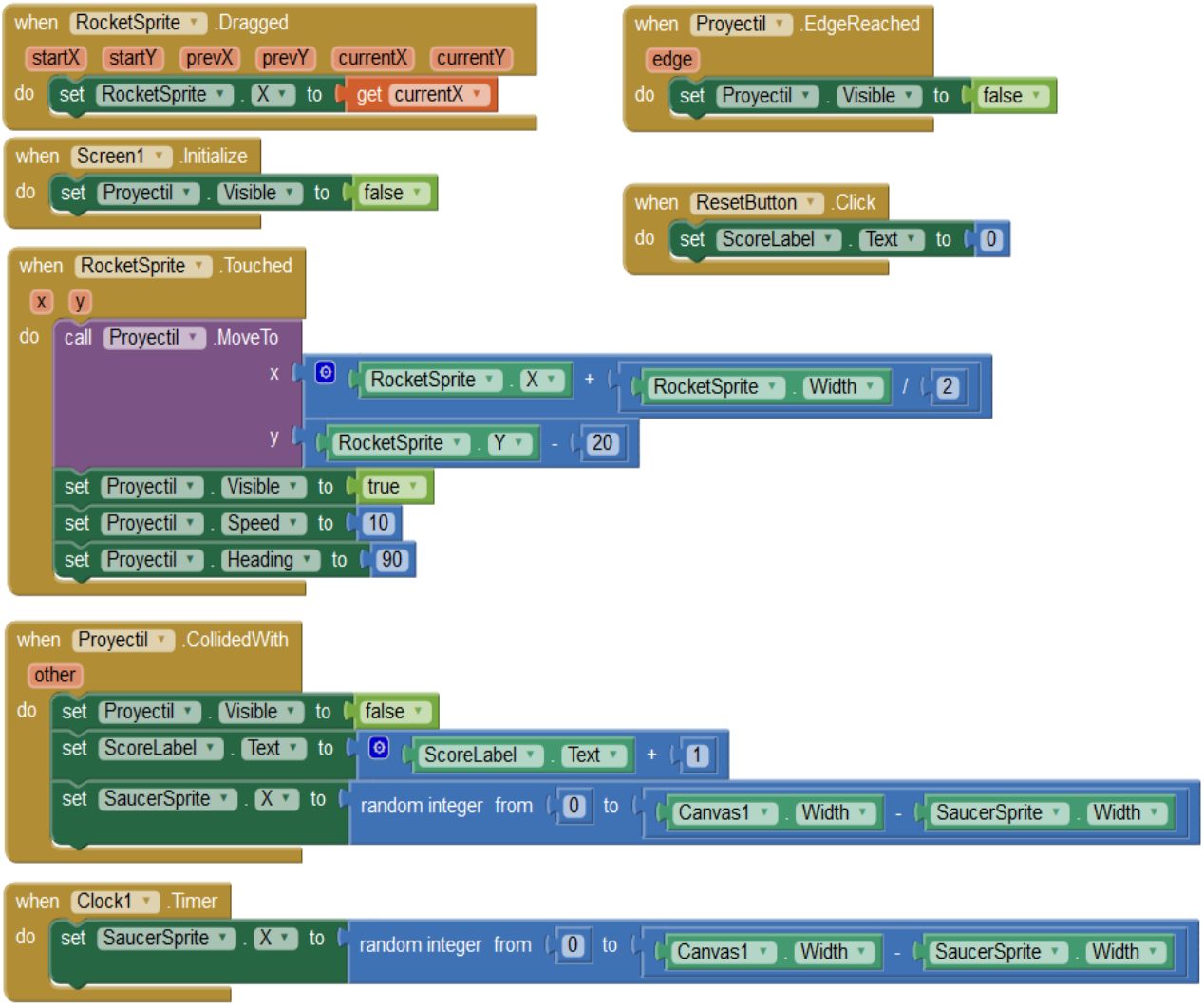
Vamos a hacer el juego un poco **más difícil !!** Ahora, cuando el **proyectil** choca con el **platillo**, vamos a **cambiar la ubicación del platillo.** El platillo mantendrá el mismo **valor Y** por lo que sólo tendremos que **cambiar la X**. Podemos hacer esto usando el **bloque de generar un número al azar.**



Para hacerlo **aún más difícil,** también cambiaremos la **posición** del platillo cuando el **temporizador** se apaga.

Programa completo

Aquí está el programa completo SpaceInvaders.



Segunda parte.

Ahora que ya sabes hacer el programa y tienes conocimientos para ampliar la jugabilidad, diseña tu propio videojuego estilo **Space Invaders.** Puedes cambiar los dibujos, velocidades, posiciones, y añadir fallos y sonidos. Demuestra que puedes ser un/a gran **"desarrollador/a de videojuegos"**