Capítulo 3

Whack A Zombie

Este capítulo muestra cómo crear un sencillo juego tipo *whack-a-mole*[1] mediante App Inventor para tu teléfono móvil Android. Este juego está inspirado en el clásico juego arcade que tiene como objetivo *machacar* con un mazo a los topos que van saliendo de sus madrigueras para conseguir la mayor cantidad posible de puntos. A través de este sencillo ejemplo, comprobarás lo fácil que es crear un juego totalmente funcional.

3.1. ¿Qué voy a construir?

La figura 3.1 muestra una captura de pantalla del juego que serás capaz de desarrollar para Android al finalizar la lectura de este capítulo. Como puedes observar, el aspecto final de este juego, bautizado como ***Whack-A-Zombie***, está personalizado para *machacar* zombies en lugar de topos[2].

[1] http://en.wikipedia.org/wiki/Whac-A-Mole

[2] Los autores del libro tenemos más simpatía por los topos que por los zombies.

Figura 3.1: Captura de pantalla del juego *Whack-A-Zombie* discutido en este capítulo y ejecutado en un teléfono con Android 2.3. En la parte superior se muestra la barra de energía y la puntuación del jugador, las cuales se actualizan cuando pulsamos sobre la pantalla. En el centro se muestra la zona de juego, en la que el zombie se va moviendo de manera aleatoria cada cierto tiempo. En la parte inferior aparece el botón de Reset para comenzar a jugar de nuevo, reseteando la puntuación y la etiqueta de energía.

En concreto, la **funcionalidad** que implementarás será la siguiente:

- Visualización de un zombie en movimiento que, de manera aleatoria, aparece en diversas posiciones de la pantalla con un intervalo de un segundo.
- Tratamiento de la interacción táctil con el juego.
  - Si consigues *machacar* al zombie, entonces el juego reproducirá un sonido, el teléfono vibrará, el zombie aparecerá en otro lugar y la puntuación se incrementará en una unidad.
  - Si fallas al *machacar* al zombie, entonces el nivel de energía se verá reducido.
- Reseteo del juego, a través de un botón, para poder jugar de nuevo.
- Actualización de la información del juego, es decir, puntuación y nivel de energía.

A través del desarrollo del tutorial de este capítulo **aprenderás a manejar los siguientes componentes**:

- Etiqueta para mostrar texto en el teléfono.
- DisposiciónHorizontal para alinear etiquetas de texto.
- SpriteImagen para imágenes dinámicas con las que interactuar a través de la pantalla del teléfono.

Tipo	Categ.	Nombre	Objetivo
Lienzo	Dibujo	Lienzo	Contenedor para el sprite del zombie
SpriteImagen	Dibujo	Zombie	Zombie a *machacar*
Reloj	Sensores	Reloj	Control del movimiento del zombie
Botón	I.U.	BotonReset	Nuevo juego
Sonido	Medios	SonidoImpacto	Reproducir sonido y vibrar cuando se golpea al zombie
Sonido	Medios	SoundFin	Reproducir sonido al terminar el juego
Etiqueta	I.U.	EtiquetaEnergia	Mostrar el texto 'Energía'
Etiqueta	I.U.	EtiquetaPuntos	Mostrar el texto 'Puntos'
Etiqueta	I.U.	EtiquetaBarra	Mostrar la barra de energía
Etiqueta	I.U.	EtiquetaPuntuacion	Mostrar el número de puntos

Tabla 3.1: Lista completa de componentes utilizados en el juego *Whack-A-Zombie*. Por cada componente se indica el tipo, la categoría en la que encontrarlo, cual será su nombre en el juego *Whack-A-Zombie*, y su objetivo (para qué se ha utilizado).

- Lienzo como superficie para colocar el componente SpriteImageSn.
- Reloj para controlar cuándo cambiar al zombie de posición.
- Sonido para reproducir sonidos y generar vibraciones en el teléfono.
- Botón para iniciar un nuevo juego.

La tabla 3.1 muestra el listado completo de componentes utilizados para desarrollar el juego *Whack-A-Zombie*.

Además, serás capaz de **implementar la siguiente funcionalidad**:

- Mover al zombie de manera aleatoria.
- Detectar cuándo se ha *machacado* el zombie y cuándo no.
- Reproducir sonido y vibraciones en tu teléfono móvil.
- Manejar contadores para la energía y la puntuación.

3.2. Diseño de la interfaz paso a paso

En esta sección aprenderás a colocar todos los componentes que forman parte del juego *Whack-A-Zombie*. Para ello, describiremos paso a paso las distintas acciones que has de realizar para obtener el aspecto del juego final que se muestra en la figura 3.1.

3.2.1. La pantalla de juego

En primer lugar, arrastra un nuevo Lienzo disponible en la paleta *Dibujo y animación*. Puedes renombrarlo a *Lienzo*. A continuación, ajusta el *Ancho* y el *Alto* al valor *Ajustar al contenedor* para que se ajuste automáticamente a la resolución de tu teléfono móvil. Ahora ya puedes añadir la imagen *background.jpg* de fondo a través del campo *ImagenDeFondo*.

El siguiente paso consiste en añadir la imagen o *sprite* asociada a nuestro zombie. Para ello, arrastra un SpriteImagen desde la paleta *Dibujo y animación*. Cámbiale el nombre por el de Zombie. Ahora ya puedes establecer la imagen del zombie a través del campo *Foto*, utilizando *zombie.png*.

Finalmente, añade un botón *Reset* que servirá para reanudar el juego una vez terminado. Para ello, arrastra un nuevo Botón desde la paleta *Interfaz de usuario* hasta justo debajo del *Lienzo*. Puedes renombrarlo a BotonReset, y cambiar el texto que aparece por defecto.

En este punto, el resultado de la interfaz de tu juego debería ser similar al que se muestra en la figura 3.2, aunque quizás la posición del zombie sea otra distinta.

3.2.2. La barra de energía y la puntuación

A continuación, vamos a añadir al juego la barra de energía y el marcador de puntuación. Estos componentes se actualizarán a medida que el juego progrese según el número de aciertos y fallos a la hora de *machacar* el zombie.

Para ello, arrastra dos componentes de tipo DisposiciónHorizontal desde la paleta *Disposición* justo por encima del componente Lienzo. Este componente, como su nombre indica, nos va a servir para alinear, de manera horizontal, otros componentes en su interior. Renombra el de la parte superior a BarraVida, mientras que el otro será BarraPuntuacion. En ambos casos, puedes dejar el ancho y el alto a valores automáticos, ya que se ajustarán a continuación con los componentes que gestionen.

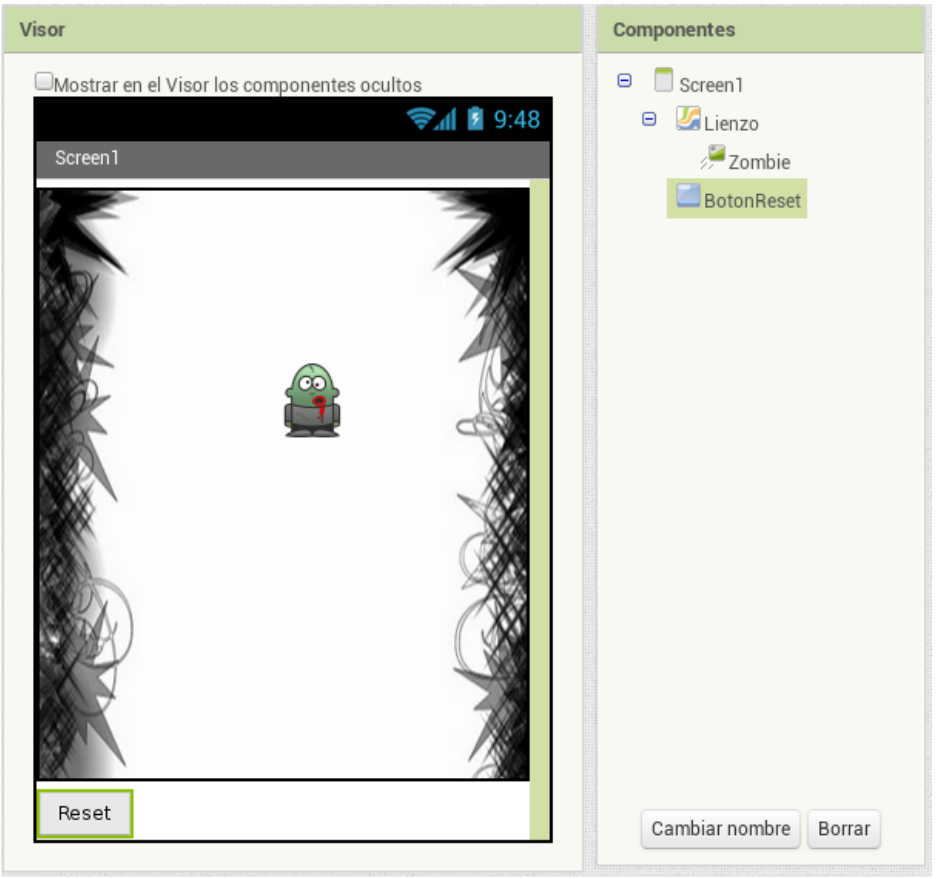

Figura 3.2: Interfaz gráfica inicial del juego *Whack-A-Zombie*.

Ahora, arrastra dos nuevas Etiquetas hasta el interior de BarraVida. La de la parte izquierda puedes renombrarla a EtiquetaEnergia, estableciendo las siguientes propiedades (parte derecha de la interfaz de App Inventor): *Negrita* activada, *Tamaño de letra* a 24.0, *TipoDeLetra* a *serif*, *Texto* a "*Energía:*" y, finalmente, *ColorDeTexto* a *Gris oscuro*.

Por otra parte, puedes renombrar la etiqueta de la parte derecha que acabas de integrar en BarraVida a EtiquetaBarra. En este caso, puedes establecer las siguientes propiedades: *ColorDeFondo* a *Azul*, *Ancho* a *100 pixeles* y *Alto* a *20 pixeles*. Borra después el contenido del campo *Texto*.

De nuevo, arrastra dos nuevas Etiquetas hacia el componente BarraPuntuacion. Para ambas, puedes utilizar las propiedades previamente establecidas para la etiqueta EtiquetaEnergia. Puedes renombrarlas

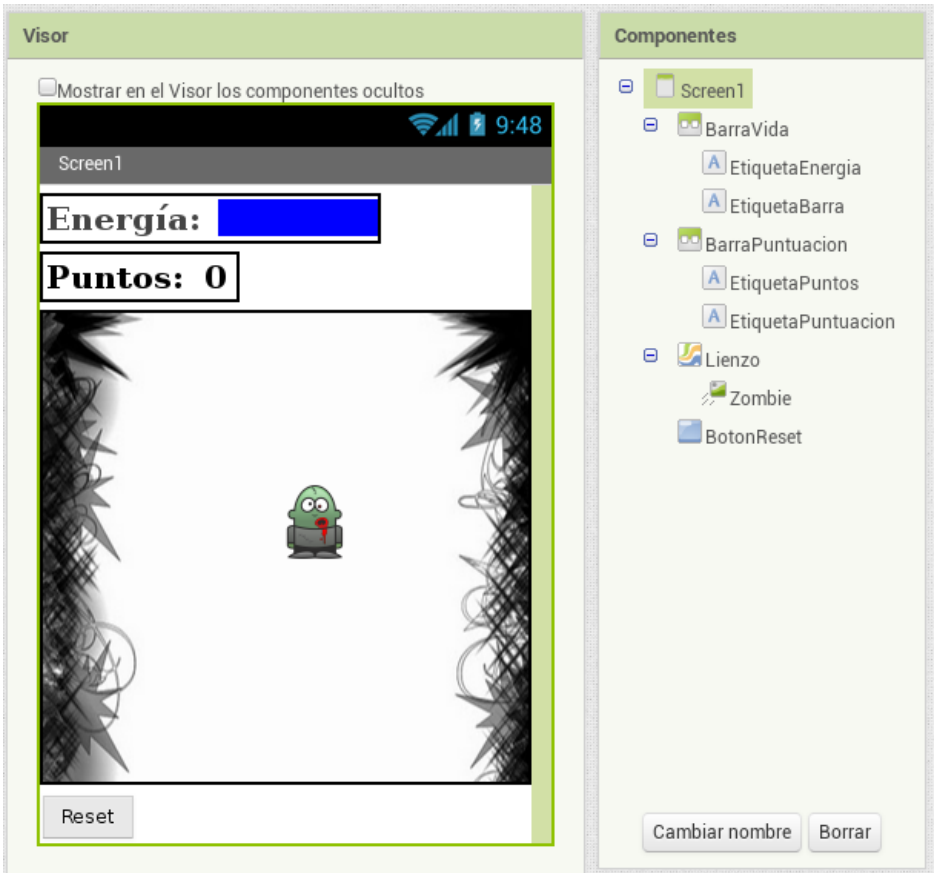

Figura 3.3: Interfaz gráfica inicial del juego *Whack-A-Zombie* integrando los componentes para la barra de energía y la puntuación.

a EtiquetaPuntos y EtiquetaPuntuacion y establecer el campo *Texto* a "*Puntos:*" y *0*, respectivamente.

En este punto, el resultado de la interfaz de tu juego debería ser similar al que se muestra en la figura 3.3.

3.2.3. El sonido y el reloj del juego

Llegados a este punto, el aspecto de la interfaz gráfica del juego ya está completado. Simplemente nos queda añadir tres componentes no visibles que servirán para reproducir sonidos en el juego y controlar cuándo éste termina.

En primer lugar, añade un nuevo Reloj desde la paleta *Sensores* y renómbralo a Reloj. Comprueba como el componente se sitúa bajo la interfaz del juego en el área de *Componentes no visibles*.

Por otra parte, añade dos nuevos Sonidos desde la paleta *Medios*. Renombra el primero a SonidoImpacto y asóciale el sonido *hit.wav* a través de la propiedad *Source*. Renombra el segundo a SonidoFin y asóciale el sonido *zombie.wav*. Estos dos componentes servirán para reproducir efectos de sonido al golpear un zombie y al terminar el juego, respectivamente.

La figura 3.4 muestra el aspecto final de la interfaz gráfica del juego *Whack-A-Zombie*. Ahora ya puedes añadir el comportamiento utilizando el editor de *Bloques*. Éste es precisamente el tema de la siguiente sección.

3.3. Defínción del comportamiento del juego

Una vez creados los distintos componentes del juego, el siguiente paso consiste en definir el comportamiento asociado a ellos, es decir, lo que debe ocurrir cuando el jugador interactúa con el juego. Básicamente, este comportamiento se puede resumir a través de los siguientes puntos:

- El zombie se moverá aleatoriamente por la pantalla de juego.
- El usuario intentará *machacar* al zombie. Ante una pulsación del jugador sobre el teléfono, pueden darse dos situaciones:
  1. Si el jugador pulsa sobre el zombie, entonces se actualiza la puntuación, se reproduce un sonido y se emite una vibración.
  2. Si el jugador falla al pulsar sobre el zombie, entonces se actualiza la barra de energía (reduciendo su tamaño).
- Una barra de energía vacía implica que el juego termina, finalizando la animación del zombie.
- El botón Reset permite reanudar el juego, es decir, establece la puntuación a 0, rellena la barra de energía y reanima al zombie.

3.3.1. Animación del zombie

Al igual que App Inventor ya proporciona funciones para, por ejemplo, reproducir un sonido mediante la función *play* del componente Sound, resulta muy interesante definir nuestras propias funciones o

procedimientos para poder utilizarlos cuando sea necesario. Un ejemplo podría ser el movimiento del zombie, es decir, podríamos crear un procedimiento que cada vez que fuera ejecutado recolocara al zombie en una nueva posición. Con esta idea en mente, hemos creado el procedimiento *MoverZombie*, cuyo diagrama de bloques se muestra en la figura 3.5.

Para poder mover el zombie correctamente, hay que tener en cuenta que su posición en la pantalla del teléfono no sea mayor que su resolución. En otras palabras, la posición de nuestro zombie en todo momento no debe estar fuera de las dimensiones del componente Lienzo que creamos previamente. La figura 3.6 muestra los valores de las coordenadas X e Y considerados para llevar a cabo el movimiento del zombie.

La construcción del diagrama de bloque se iniciará arrastrando un bloque de tipo **como procedimiento ejecutar** de la categoría *Procedimientos*, tal y como se muestra en la figura 3.5. Renómbralo a a *MoverZombie*. Ahora ya puedes integrar una llamada al bloque **llamar Zombie.MoverA**, que encontrarás al seleccionar dentro del área de bloques al componente Zombie.

En este punto, ya tenemos disponible la **base para mover el zombie** a una nueva posición. Ahora es necesario, precisamente, generar de manera aleatoria dicha posición, es decir, es necesario generar un nuevo valor para las coordenadas X e Y del zombie, teniendo en cuenta que no sobrepasen las dimensiones de la pantalla.

Haz *clic* sobre la categoría *Matemáticas* dentro de los bloques *Integrados*. Ensambla ahora un bloque del tipo **entero aleatorio entre y** al parámetro *x* de **Zombie.MoverA**. El valor aleatorio de la nueva posición en X del zombie ha de ser un entero comprendido entre 1 y la anchura del canvas (restando el tamaño del zombie para evitar que pueda quedar oculto). Para ello, ensambla un bloque del tipo **número** (también *Matemáticas* en el parámetro **entre** del **entero aleatorio** que acabas de añadir y asígnale un valor de 1.

Por otra parte, ensambla un bloque de resta, como se muestra en la figura 3.5, donde la primera parte tenga la anchura del canvas y la segunda la anchura del zombie en la rama *y* del **entero aleatorio**.

Finalmente, para generar la coordenada Y del zombie puedes seguir el mismo esquema, utilizando simplemente la altura del Lienzo y del Zombie. Deberías obtener un resultado similar al que se muestra en la figura 3.5.

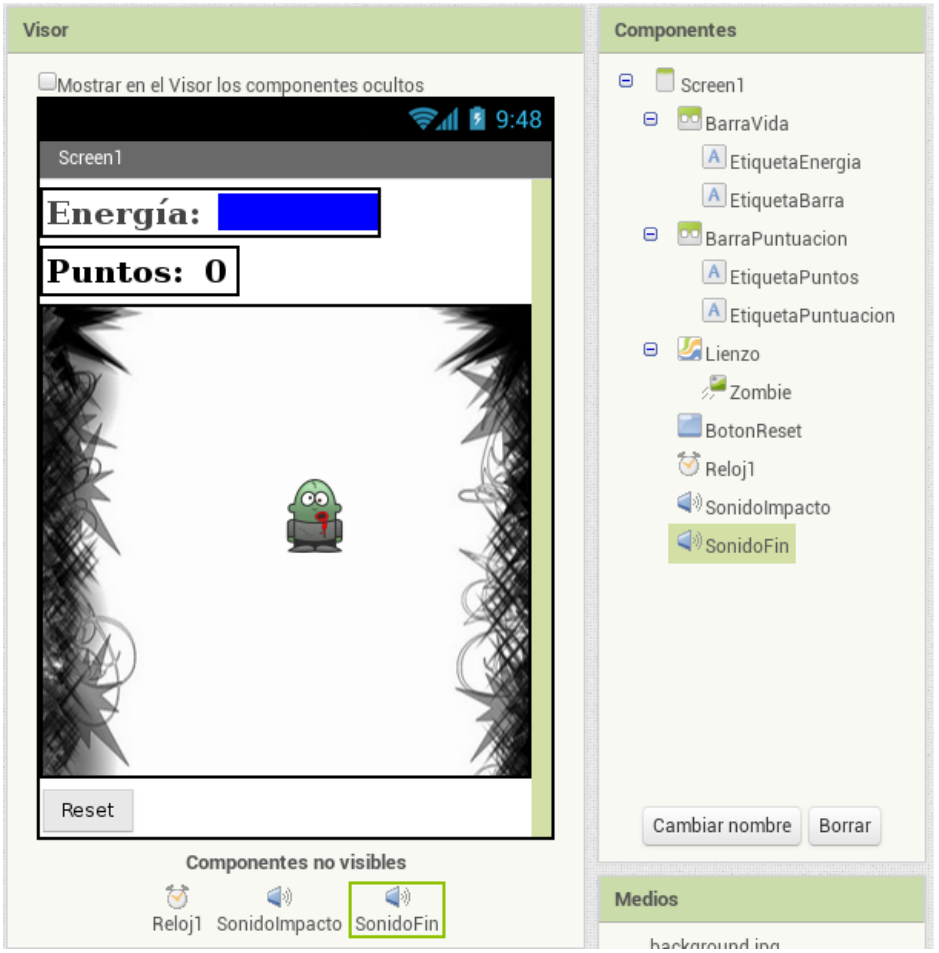

Figura 3.4: Aspecto final del diseño del juego *Whack-A-Zombie*.

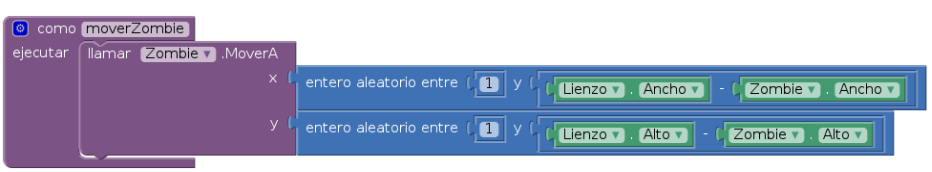

Figura 3.5: Procedimiento *MoverZombie* para colocar el zombie, de manera aleatoria, en distintas posiciones de la pantalla.

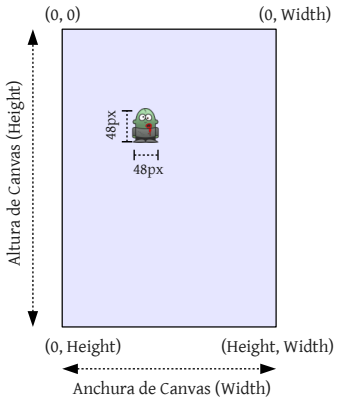

Figura 3.6: Representación gráfica de la posición del zombie dentro del Lienzo. Las dimensiones de este componente están representadas por su *Ancho* y *Alto*. La coordenada de la parte superior izquierda del Lienzo está representada por el valor $(0, 0)$, es decir, 0 unidades en el eje X y 0 unidades en el eje Y. La coordenada de la parte inferior derecha está representada por el valor $(Ancho, Alto)$, es decir, *Ancho* unidades en el eje X y *Alto* unidades en el eje Y.

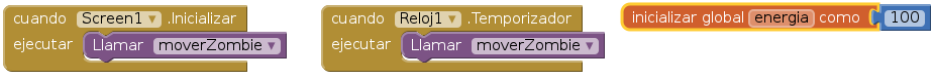

Figura 3.7: Arranque de la aplicación y asignación de valores iniciales.

3.3.2. Arranque del juego

Una vez modelado el bloque para mover al zombie, ya es posible utilizarlo. En este contexto, el comportamiento esperado del juego que estás desarrollando se basa en que el zombie comience a moverse justo al arrancar la aplicación. Para ello, utiliza el bloque **Screen1.Inicializar**, tal y como se muestra en la parte izquierda de la figura 3.7 (búscalo haciendo clic sobre la categoría *Screen1* dentro del área de bloques. Después, ensambla en este bloque el procedimento **MoverZombie**, disponible en la categoría *Procedimientos*.

Además, tienes que asegurarte que **el zombie continúe moviéndose** sin parar cada cierto tiempo sobre la pantalla mediante el componente Reloj definido anteriormente. Este componente tenía por defecto un valor de intervalo de 1000 milisegundos, es decir, 1 segundo[3]. Así, el zombie cambiará de posición cada segundo, dificultando de esta manera el *golpeo* por parte del jugador. Este comportamiento se especifica fácilmente mediante el componente **Reloj.Temporizador**. Ahora, tan sólo tienes que ensamblar en el mismo el bloque **MoverZombie** que definiste previamente. Justo en este punto puedes apreciar la utilidad de **MoverZombie**. El resultado debería ser similar al que se muestra en la parte central de la figura 3.7.

Por otra parte, también suele ser común establecer unos **valores de inicio** para algunos elementos en función de la aplicación que se está desarrollando. En el caso del *Whack-A-Zombie*, tienes que definir la cantidad de energía o vida inicial. Esta energía disminuirá cuando falles al golpear al zombie. Para definir la variable energia, ve a la categoría *Variables* y arrastra un bloque **inicializar global como**. A continuación, si haces clic sobre **variable**, podrás renombrarla a *energia*. Ahora, sólo falta asignarle el valor inicial, que será de 100. Puedes hacerlo a partir del bloque **número** de la categoría desde *Matemáticas*, o simplemente tecleando el valor 100 y pulsando el retorno de carro sobre una zona vacía de la pantalla. El resultado esperado se muestra en la parte derecha de la figura 3.7.

[3]Puedes ajustar este valor para que el zombie se mueva tan rápido como quieras.

3.3.3. Interacción con el juego

Como recordarás de la sección 3.2, creaste dos etiquetas, nombradas EtiquetaPuntuacion y EtiquetaBarra, para mostrar al jugador de *Whack-A-Zombie* los puntos que va acumulando y la cantidad de energía restante. ¡Recuerda que si el jugador consume su energía, entonces el juego terminará!

En este punto vamos a definir la lógica necesaria para **actualizar las etiquetas** en función de la interacción del usuario con el juego. Para ello, usarás el bloque **cuando Lienzo.Tocar**, que indica que el jugador tocó la pantalla, las coordenadas X e Y del toque (para nosotros son irrelevantes) y si alguno de los *sprites*, como nuestro zombie, fue tocado (esto sí que nos interesa). La figura 3.8 muestra el código necesario para actualizar las dos etiquetas mencionadas anteriormente y para reproducir un efecto sonoro al terminar el juego. No te preocupes por la longitud del bloque, ya que lo vamos a analizar paso a paso.

Como hemos comentado, el bloque **Lienzo.Tocar** comprueba si el toque alcanzó algún *sprite* o imagen del juego, como nuestro zombie, cuando se interactúa con la pantalla del móvil. Debido a que en *Whack-A-Zombie* sólo tenemos un *sprite* (Zombie), entonces sólo existe la opción de alcanzar a éste. **Si el jugador pulsa sobre el *sprite* Zombie**, entonces debería ocurrir lo siguiente:

- La puntuación se incrementará en 1.
- El juego reproducirá un sonido para que el jugador sepa que ha alcanzado al zombie.
- El teléfono móvil vibrará para que el jugador tenga algo más de realimentación.

Para modelar este comportamiento, puedes arrastrar el bloque **Lienzo.Tocar**. El resultado debería ser similar al de la parte superior de la figura 3.8.

A continuación, realiza los siguientes pasos:

1. Ensambla en la rama *ejecutar* de **Lienzo.Tocar** un bloque de control del tipo **si entonces**. Este bloque nos servirá para controlar que el juego continúe sólo si el jugador tiene energía. En este punto, ¡ya deberías dominar a la perfección dónde se encuentran dichos bloques!
2. Ensambla en la rama de comprobación de la condición del anterior bloque de control un bloque $>$ (mayor que). Este bloque lo vamos

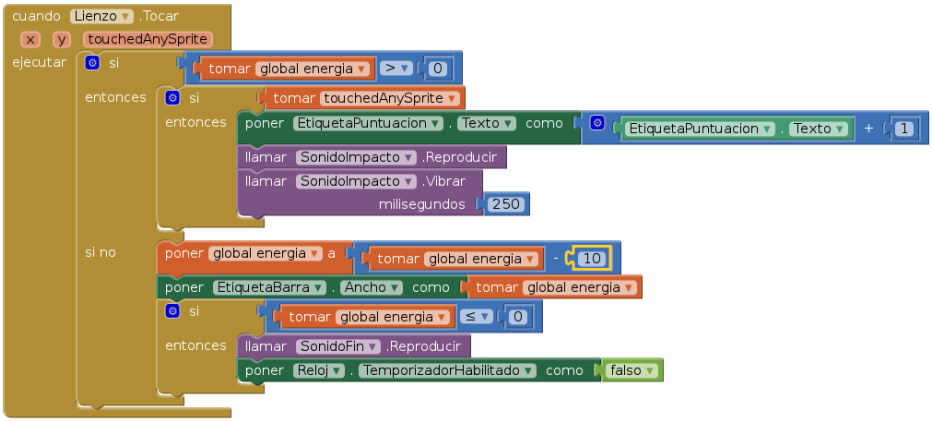

Figura 3.8: Interacción con la pantalla del juego. Control de energía y actualización, en su caso, de la puntuación.

a utilizar para comprobar si el nivel de energía, (es decir, la variable energia es mayor que 0. Ensambla el bloque **tomar global energia** desde *Variables* en la primera parte de dicho bloque. Ahora ensambla un 0 en la segunda parte.

3. Ensambla en la rama **entonces** otro bloque del tipo **si entonces**. Este bloque nos va a servir para definir el comportamiento en caso de que acertemos al golpear al zombie o, en caso contrario, que fallemos. Para ello, ensambla en la rama junto al **si** el bloque **tomar touchedAnySprite**, que te aparecerá al situar el cursor sobre el argumento del mismo nombre que encontrarás en la cabecera del bloque.

Antes de pasar a añadir el código asociado a golpear o fallar sobre el zombie, deberías comparar tu código con el de la figura 3.8 para comprobar que todo es correcto.

Ahora ya podemos añadir el código relativo al **golpeo del zombie**. Para ello, sigue los siguientes pasos:

1. Ensambla el bloque **poner EtiquetaPuntuacion.Texto como**, en la rama **entonces** del anterior bloque condicional. Completa el bloque, ensamblando una suma (en categoría *Matemáticas*, o simplemente pulsando + sobre una zona vacía del área de trabajo), y encaja como términos **EtiquetaPutuación.Texto** y **1**. Esto nos permite incrementar en 1 la puntuación cuando se golpea al zombie.

2. Ensambla el bloque **SonidoImpacto.Reproducir** para que se reproduzca el sonido que previamente asociamos a éste en la sección 3.2.

3. Ensambla el bloque **SonidoImpacto.Vibrar** con un valor de 250 en la rama *milisegundos*. De este modo, el teléfono vibrará durante un cuarto de segundo cuando golpees al zombie.

La otra cara de la moneda está representada por el código relativo al **fallo a la hora de golpear al zombie**. Para ello, sigue los siguientes pasos:

1. pulsa en el cuadrado azul del primer bloque **si entonces** para añadir la rama del **si no**, ya que esta no aparece por defecto. Al hacerlo verás un bocadillo con dos partes. Arrastra el bloque **si no** de la izquierda hasta el de la izquierda, encajándolo dentro del **si**. Ahora el condicional debería incluir las dos ramas.

2. Ensambla un bloque **poner global energia a** desde *Variables* en la rama **si no** del bloque condicional y asóciale una resta donde el primer término sea la variable energia y el segundo el número 10. Este bloque permite reducir el valor de la variable energia.

3. Ensambla un bloque **poner EtiquetaBarra.Ancho a** y asígnale el valor que tenga la variable energia, tal y como se muestra en la figura 3.8. Así, la longitud de la barra de energía se actualizará conforme el valor de la variable vaya cambiando.

La última parte de la rama **si no** del bloque condicional nos va a servir para controlar la **parada del juego**. Éste debería terminar cuando el jugador se quede sin barra de energía. En otras palabras, el juego debería acabar cuándo la variable energia alcance un valor igual o inferior a 0. Para ello:

1. Ensambla un bloque **si entonces** y añade como condición un bloque <= que permita comparar el valor de energia y el número 0.

2. Ensambla un bloque **SonidoFin.Reproducir** en la rama **entonces**. De este modo, el juego reproducirá un sonido cuando finalice.

3. Ensambla un bloque **poner Reloj.TemporizadorHabilitado como** y asígnale un valor **falso** desde la categoría *Lógica*. Esto permite detener el movimiento del zombie, ya que éste está controlado por un valor *cierto* en el bloque que se muestra en la figura 3.7. Si establecemos este valor a falso, entonces la rama **ejecutar** de dicho bloque, es decir, la que ejecuta *MoverZombie* no se ejecutará y, por lo tanto, el zombie no se moverá.

En este punto, ya deberías tener un juego completamente funcional y tu código debería ser similar al de la figura 3.8. La última parte que nos queda por discutir es el *reset* del juego para dar lo opción al jugador de seguir jugando a *Whack-A-Zombie* todas las veces que quiera.

3.3.4. Reinicio juego

Con el objetivo de realizar un *reset* de *Whack-A-Zombie* es necesario ejecutar las siguientes acciones:

- Reestablecer la energía del jugador al máximo.
- Poner la puntuación a 0.
- Volver a poner al zombie en movimiento.

Figura 3.9: Reseteo del juego e interacción con el zombie (sonido de golpe y vibración del teléfono).

La figura 3.9 muestra el código necesario para efectuarlas. Este código se ejecutará cuando el jugador pulse el botón Reset que está en la parte inferior del juego, tal y como se indica en la figura 3.1 al inicio de este capítulo. Para añadir este último bloque, sigue los siguientes pasos:

1. Añade un bloque **cuando BotonReset.Clic**. Precisamente, la palabra *Clic* hace referencia a pulsar dicho botón.

2. Ensambla el bloque **poner global energia** y asígnale un valor de 100, exactamente igual que en la inicialización de la sección 3.3.2.

3. Ensambla el bloque **poner EtiquetaBarra.Ancho** al valor de energía, exactamente igual que en la sección 3.3.3.

4. Ensambla el bloque **poner EtiquetaPuntuacion.Texto** con un valor de 0.

5. Ensambla el bloque **poner Reloj.TemporizadorHabilitado** a *cierto* (*Built-in->Logic*). Esto permite que el zombie empiece a moverse de nuevo, exactamente igual que al inicio de la aplicación.

En este punto, deberías comparar tu código con el que se muestra en la figura 3.9. Ahora, ya puedes probar tu juego y disfrutar de él en tu teléfono móvil.