CAPÍTULO CINCO

PROGRAMACIÓN

Education and Culture DG
Lifelong Learning Programme

This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Programa de aprendizaje permanente

APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Indice

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

1. INTRODUCCION AL CORONA SDK: DESARROLLO DE EASY CROSS-PLATFORM

Introducción

Corona SDK es una excelente opción para todo tipo de desarrolladores móviles desde principiantes hasta expertos. Este tutorial le dará a conocer esta multiplataforma marco fácil de usar,y le mostrará cómo comenzar a crear contenido para su plataforma preferida.

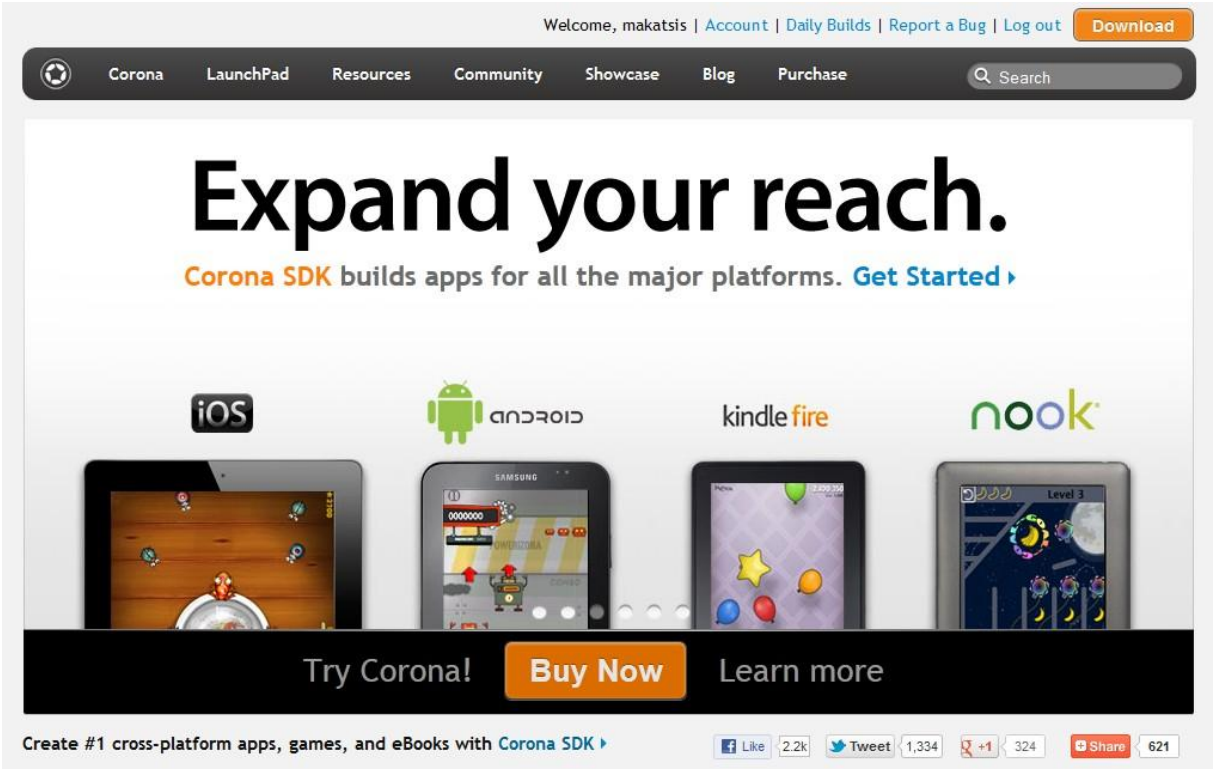


Figura 1: Sitio Web Oficial de Corona

La web oficial Corona describe el SDK de la siguiente manera:

"Corona es una herramienta de desarrollo rápido y fácil para los juegos de iPhone, iPad y Android y aplicaciones. Corona-powered apps ejecuta aplicaciones a 30 fps en tan poco como 300k, y los gráficos y el motor de animación aprovecha plenamente la aceleración del hardware OpenGL.

El Corona SDK es el primero de la familia Corona Ansca de productos para la creación de alto rendimiento multimedia de aplicaciones gráficamente ricas y juegos para el iPhone. Con Corona, usted puede crear rápidamente aplicaciones para el iPhone en cuestión de horas. No se requiere Objetivo-C/**Cocoa** ni **C++.**

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

Ansca es la compañía detrás de Corona. Este SDK permite a los desarrolladores crear aplicaciones multiplataforma rápidas y potentes que tienen acceso a API no es así con otros marcos, como la cámara, GPS y acelerómetro.

Corona SDK ofrece un montón de características que lo convierten en una forma muy fiable para crear aplicaciones. Algunas de estas características son:

• **Desarrollo de aplicaciones nativas**: Los binarios ejecutables Corona son 100% Objective-C / C + +, por lo que no tendrá que preocuparse de las nuevas reglas Apple iOS5 sobre el uso de las herramientas de desarrollo exterior. De hecho, Corona necesita Xcode para compilar.

• **Integración Automática OpenGL-ES**: No hay necesidad de llamar a las clases o funciones extensas para crear simples manipulaciones de pantalla.

• **El desarrollo multiplataforma**: Corona puede crear aplicaciones para iOS (iPhone, iPod Touch, iPad) y mecanismos Android.

• **Rendimiento**: Corona está optimizado para hacer uso de las características aceleradas por hardware, consiguiendo un potente rendimiento en juegos y aplicaciones.

• **Características del dispositivo**: Controles de acceso nativos de los dispositivos y hardware, como la cámara, acelerómetro, GPS, etc.

• **Fácil de aprender**: Corona utiliza el lenguaje de programación Lua (**Lua programming language)**, que es potente y fácil de aprender.

Plataformas de apoyo

La mayor ventaja de la Corona es que le permiten trabajar con una base de código y producir productos para muchos dispositivos diferentes.

En concreto, el SDK Corona le permitirá crear aplicaciones para todos los dispositivos **iOS** y **Android**.

Desarrollo con Lua

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Figura 2: El lenguaje de programación Lua

Corona utiliza el **lenguaje de programación Lua** para crear aplicaciones. Lua es un lenguaje de secuencias de comandos comúnmente utilizado para desarrollar juegos. Tiene una buena cantidad de adopción de mercado en el desarrollo de la comunidad. Sintaxis Lua puede ser comparado con lenguajes como **JavaScript** o **ActionScript 3**, que hace que sea fácil de aprender.

En la actualidad, muchos lenguajes de programación tienen que ver con la manera de ayudar a escribir programas con cientos de miles de líneas. Por eso, te ofrecemos paquetes, espacios de nombres, sistemas de tipo complejo, una multitud de construcciones, y miles de páginas de documentación para ser estudiadas.

Lua no trata de ayudar a escribir programas con cientos de miles de líneas. En su lugar, Lua intenta ayudar a solucionar su problema con sólo cientos de líneas, o incluso menos. Para lograr este objetivo, Lua se basa en la extensibilidad, al igual que muchos otros idiomas. A diferencia de la mayoría de los otros idiomas, sin embargo, Lua se puede extender fácilmente no sólo con software escrito en el mismo Lua, sino también con el software escrito en otros lenguajes, como **C** y **C + +**.

Lua fue diseñado, desde el principio, para integrarse con software escrito en C y otros lenguajes convencionales. Esta dualidad de idiomas trae muchos beneficios. Lua es un lenguaje pequeño y sencillo, en parte porque no trata de hacer lo que en C ya es bueno, como el rendimiento puro, operaciones de bajo nivel o interfaz con el software de terceros. Lua se basa en C para esas tareas. Lo que Lua ofrece es en lo que C no es bueno: una buena distancia del hardware, estructuras dinámicas, ninguna redundancia, la facilidad de pruebas y depuración. Para ello, Lua tiene un ambiente seguro, gestión de memoria automática, y gran facilidad para manejar cadenas y otros tipos de datos con tamaño dinámico.

Más que ser un lenguaje extensible, Lua es un lenguaje de pegamento. Lua soporta un enfoque basado en componentes para el desarrollo de software, donde creamos una aplicación pegando existentes componentes de alto nivel. Por lo general, estos componentes están escritos en un compilado, lenguaje de tipos estáticos, como C o C + +, Lua es el pegamento que se utiliza para componer y conectar esos componentes. Por lo general, los componentes (u objetos) representan conceptos más concretos de bajo nivel (como los widgets y las estructuras de datos) que no están sujetos a muchos cambios durante el desarrollo del programa y que se llevan la mayor parte del **CPU time** del programa final. Lua da la forma final de la aplicación, lo que probablemente va a cambiar mucho durante el ciclo de vida del producto. Sin embargo, a diferencia de otras tecnologías de la cola, Lua también es un lenguaje de pleno derecho. Por lo tanto, podemos usar Lua no sólo para los

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

componentes del pegamento, sino también para adaptarse y remodelarse a ellos, o incluso para crear nuevos componentes integrales.

Por supuesto, Lua no es el único lenguaje de *scripting*. Hay otros idiomas que se pueden utilizar para los mismos fines más o menos, tales como Perl, Tcl, Ruby, Forth y Python. Las características siguientes sitúan a Lua fuera de estas lenguas, aunque otros idiomas comparten algunas de estas características con Lua, ninguna otra lengua ofrece un perfil similar:

Extensibilidad: La extensibilidad de Lua es tan sorprendente que muchas personas consideran Lua no como un lenguaje, sino como un kit para la construcción de lenguajes específicos de dominio. Lua ha sido diseñado desde cero para extenderse, tanto a través de código Lua y a través de código externo C. Como prueba de concepto, se aplica la mayor parte de su funcionalidad básica propia a través de librerías externas. Es realmente fácil de interfaz Lua con C / C + + y otros lenguajes, como **Fortran**, **Java**, **Smalltalk**, **Ada**, e incluso con otros lenguajes de scripting.

• **Simplicidad**: Lua es un lenguaje sencillo y pequeño. Tiene pocos (pero potentes) conceptos. Esta simplicidad hace que Lua sea fácil de aprender y contribuye a una aplicación pequeña. Su distribución completa (código fuente, manual, además de los binarios para algunas plataformas) encaja cómodamente en un disquete.

• **Eficiencia**: Lua tiene una implementación muy eficiente. Puntos de referencia independientes muestran Lua como una de las lenguas más rápidas en el ámbito de *scripting* (interpretado) de idiomas.

• **Portabilidad**: Cuando hablamos de portabilidad, no estamos hablando acerca de la ejecución de Lua en Windows o en Unix. Estamos hablando acerca de la ejecución de Lua en todas las plataformas que hemos escuchado sobre: **NextStep, OS / 2, PlayStation II (Sony), Mac OS-9** y **OS X, BeOS, MS-DOS**, **IIBM mainframes**, **EPOC**, **PalmOS**, **RISC OS** , además de, por supuesto, todos los sabores de Unix y Windows. El código fuente de cada una de estas plataformas es prácticamente el mismo. Lua no utiliza la compilación condicional para adaptar su código a diferentes máquinas, sino que se adhiere a la norma **ANSI (ISO) C**. De esta manera, por lo general no es necesario adaptarse a un nuevo entorno: Si usted tiene un compilador ANSI C, sólo tiene que compilar Lua, fuera de la caja.

Gran parte del poder de Lua proviene de sus bibliotecas. Esto no es por casualidad. Una de las principales fortalezas de Lua es su extensibilidad a través de nuevos tipos y funciones. Muchas características contribuyen a este punto fuerte. La escritura dinámica permite un alto grado de polimorfismo. La gestión de memoria automática simplifica las interfaces, ya que no hay necesidad de decidir quién es el responsable de asignar y desasignar memoria, o cómo manejar desbordes. Las funciones de orden superior y funciones anónimas permiten un alto grado de parametrización, haciendo las funciones más versátiles.

Lua viene con un pequeño conjunto de bibliotecas estándar. Al instalar Lua en un entorno fuertemente limitado, como procesadores integrados, puede ser conveniente elegir cuidadosamente las bibliotecas que se necesitan. Por otra parte, si las limitaciones son difíciles, es fácil ir dentro del código fuente de las bibliotecas y elegir una a una las funciones que se deben mantener. Recuerde, sin embargo, que Lua es bastante pequeño (incluso con todas las bibliotecas estándar) y en la mayoría de los sistemas se puede utilizar todo el paquete sin ningún tipo de preocupaciones.

Editores Lua

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

En este momento, Corona no viene con un exclusivo editor Lua, pero hay algunos grandes editores ya disponibles y que se pueden utilizar:

Gratis:

- **Eclipse**, usando el plugin de Eclipse Lua.
- **LuaEdit**, *LuaEdit* es un editor IDE / Depurador / script diseñado para la versión 5.1 de Lua.
- **Notepad + +**, un editor gratuito de código fuente que soporta varios lenguajes de programación, incluyendo Lua.
- **TextWrangler**, un poderoso editor de texto general y de Uni y herramienta de servidor de administrador.

Comercial:

- **TextMate**, disponible para Mac OS X solamente.
- **BBedit**, un profesional líder en HTML y editor de texto para el Macintosh.
- **Decoda**, un entorno de desarrollo profesional para la depuración de scripts Lua en las aplicaciones.

Creación de su primer programa

Para empezar con Corona, vamos a comenzar con la aplicación Hello World clásica.
Abra su editor preferido Lua y escriba el siguiente código: `print("Hello World!").`

Cree una nueva carpeta de proyecto denominado *HelloWorld* y guarde el archivo como *main.lua.*
Vamos a lanzar esta aplicación en los siguientes pasos.

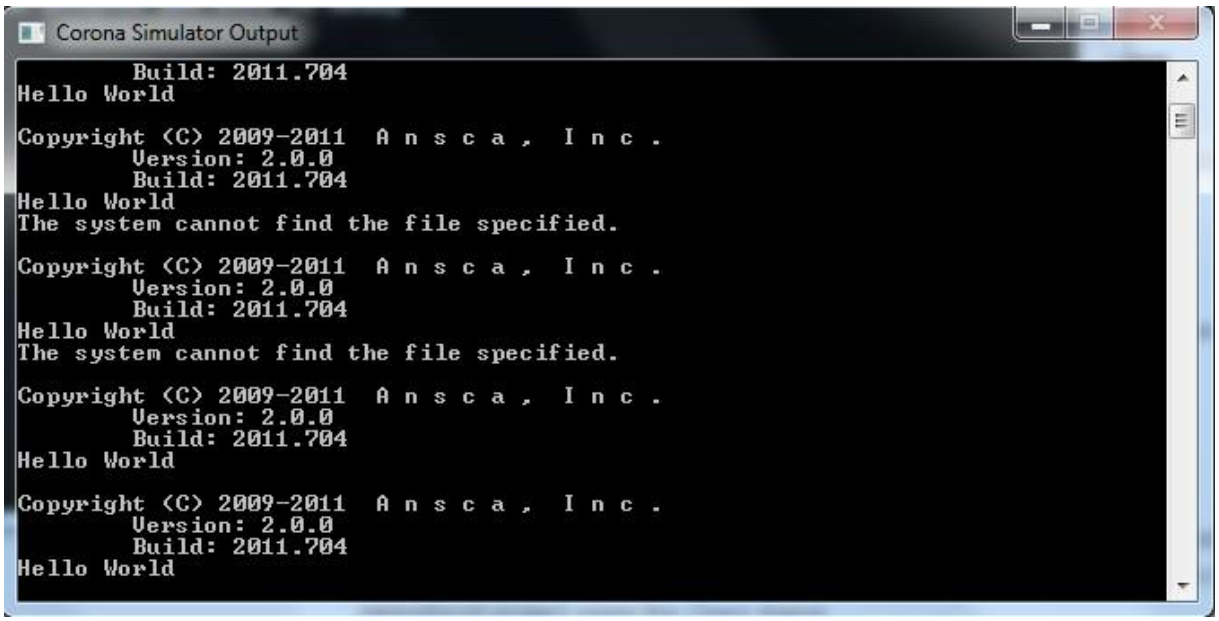


Figura 3: Terminal Corona

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Esto también se abrirá en el *simulador Corona* mostrando un gráfico iPhone sin contenido, esto se debe a la función de impresión, sólo produce la salida del terminal, para ver cómo mostrar el texto en el simulador continue con el siguiente paso.

Simulador

Para acceder al simulador o dispositivo de pantalla actual, tendremos que hacer uso de la **Corona específico de la API`s**

En el archivo main.lua escribir lo siguiente y a continuación, ejecute de nuevo el programa:

```
local myTextField = display.newText("Hello World!", 1, 20, nil,
14);myTextField:setTextColor(255, 255, 255);
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Figura 4: Simulador de Corona

2. CORONA SDK: CREACIÓN DE UN RELOJ ANALOGICO APP

Uso de la API de Corona, vamos a crear un reloj analógico básico. Los gráficos serán exportados de PNG desde el editor de imágenes que elija y después alimentados por Lua. También aprenderá cómo probar la aplicación utilizando el simulador y construyendo su aplicación para las pruebas del dispositivo.

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Selección del dispositivo de destino

Lo primero que tiene que hacer es seleccionar la plataforma en la que desea ejecutar su aplicación, de esta manera podrá elegir el tamaño de las imágenes que se van a utilizar.

La plataforma **iOS** tiene las siguientes características:

- **iPad**: 1024x768px, 132 ppi
- **iPhone / iPod Touch**: 320x480px, 163 ppi
- **iPhone 4**: 960x640px, 326 ppi

Para **Android**, es un poco diferente, al ser una plataforma abierta, usted puede encontrar muchas resoluciones de pantalla diferentes:

- **Nexus One**: 480x800px, 254 ppi
- **Droid**: 854x480px, 265 ppi
- **HTC Legend**: 320x480px, 180 ppi

Interfaz

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Figura 5: Interfaz del reloj

Código

Fondo

Lo primero que vamos a hacer es agregar el fondo del reloj:

```
local background = display.newImage("background.png")
```

Esta línea crea el fondo variable local y utiliza la API de pantalla para añadir la imagen específica en el escenario. Por defecto, la imagen se añade a 0,0.

Mostrar las manecillas de reloj

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

Repetimos el proceso con las manecillas del reloj y las imágenes del centro del reloj, colocándolas en el centro de la escena:

```
local hourHand = display.newImage("hourHand.png", 152, 185)
local minuteHand = display.newImage("minuteHand.png", 152, 158)
local center = display.newImage("center.png", 150, 230)
local secondHand = display.newImage("secondHand.png", 160, 155)
```

Punto de referencia

Para colocar correctamente las imágenes, modificamos el punto de referencia con el fin de mover imágenes relativamente a la parte inferior central:

```
hourHand:setReferencePoint(display.BottomCenterReferencePoint)
minuteHand:setReferencePoint(display.BottomCenterReferencePoint)
secondHand:setReferencePoint(display.BottomCenterReferencePoint)
```

Posición inicial

Aquí establecemos la posición inicial de las manecillas del reloj. Esta vez se establece la rotación de acuerdo con la hora del sistema:

```
local timeTable = os.date("*t")
hourHand.rotation = timeTable.hour * 30 + (timeTable.min * 0.5)
minuteHand.rotation = timeTable.min * 6
secondHand.rotation = timeTable.sec * 6
```

Prácticas de la memoria

La variable *timeTable* se utilizará sólo una vez en el inicio de la aplicación, así que no hay necesidad de guardarlo en la memoria. Para liberar la memoria utilizada por la variable (que es casi nada, pero debe acostumbrarse a desasignar objetos no utilizados) que establezca su valor a cero, esta forma de recolección de basura se encarga de ello:

```
timeTable = nil
```

Función del movimiento de las agujas

Las siguientes líneas de código manejan la rotación del reloj, es el mismo código que antes, sólo que esta vez envuelto en una función que se ejecutará cada segundo por un temporalizador.

```
local function moveHands(e)

	local timeTable = os.date("*t")
	hourHand.rotation = timeTable.hour * 30 + (timeTable.min * 0.5)
	minuteHand.rotation = timeTable.min * 6
	secondHand.rotation = timeTable.sec * 6
end
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

El temporalizador

El temporizador, se ejecuta cada segundo y realiza la función especificada, se trata de la función *moveHands* que hemos creado en el paso anterior. Las veces que se ejecutan son fijados por el tercer parámetro, 0 es infinito.

Figura 6: Reloj analógico Corona Simulador

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

3. ACELERÓMETRO. RESUMEN DE APLICACIONES

Utilizando el **Corona API's**, vamos a crear una aplicación básica que registra el dispositivo de movimiento en función del valor del acelerómetro, moviendo un objeto en la pantalla.

Selección del dispositivo de destino

Lo primero que tiene que hacer es seleccionar la plataforma en la que desea ejecutar su aplicación, de esta manera podrá elegir el tamaño de las imágenes que se van a utilizar. La plataforma **iOS** tiene las siguientes características:

- **iPad**: 1024x768px, 132 ppi
- **iPhone/iPodTouch**: 320x480px, 163 ppi
- **iPhone 4**: 960x640px, 326 ppi

Para **Android**, es un poco diferente, al ser una plataforma abierta, usted puede encontrar muchas resoluciones diferentes de pantalla:

- **Nexus One**: 480x800px, 254 ppi
- **Droid**: 854x480px, 265 ppi
- **HTC Legend**: 320x480px, 180 ppi

Interfaz

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

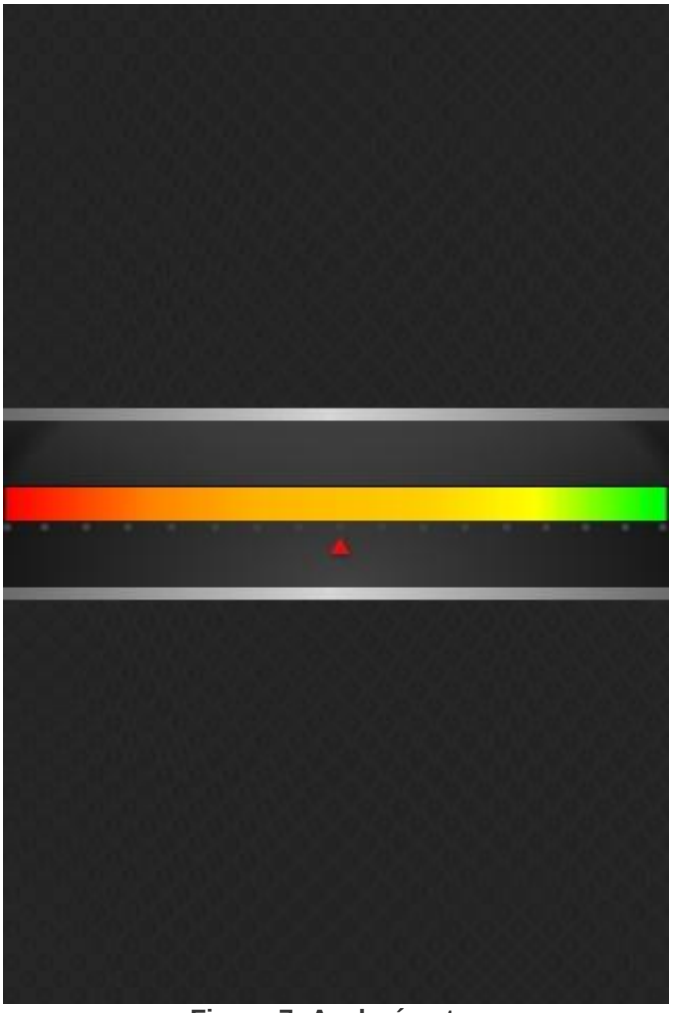


Figura 7: Acelerómetro

Esta es la interfaz gráfica que va a utilizarse, se incluye un gráfico triangular que servirá como el medidor de posición.

Código

Ocultar barra de estado

En primer lugar, ocultar la barra de estado, esto es la barra en la parte superior de la pantalla que muestra la hora, la señal, y otros indicadores.

```
display.setStatusBar(display.HiddenStatusBar)
```

Fondo

Ahora añadimos el fondo app.

```
local background = display.newImage("background.png")
```

This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Esta línea crea el fondo variable local y utiliza la API de pantalla para añadir la imagen especificada en el escenario. Por defecto, la imagen se añade a 0,0 con la esquina superior izquierda como punto de referencia.

Indicador

Se repite el proceso con el indicador de posición de imagen, colocándolo en el centro del escenario.

```
local indicator = display.newImage("indicator.png")
indicator:setReferencePoint(display.CenterReferencePoint)
indicator.x = display.contentWidth * 0.5
indicator.y = display.contentWidth * 0.5 + 100
```

Variables necesarias

Las siguientes variables se utilizan para controlar el asunto del acelerómetro.

- **acc**: una tabla que se utiliza como una función oyente para el asunto del acelerómetro.
- **CenterX**: Almacena el valor central horizontal de la escena.

```
local acc = {}
local centerX = display.contentWidth * 0.5
```

Función del Acelerómetro

Esta función utiliza la tabla de acc para crear un detector para el caso del acelerómetro, la propiedad *xGravity* (parte del evento acelerómetro) y la variable *centerX* desplaza el indicador de posición de acuerdo a la posición calculada.

```
function acc:accelerometer(e)
	indicator.x = centerX + (centerX * e.xGravity)
end
```

Esto hará que nuestro indicador equilibre cuando la inclinación del dispositivo cambie, la propiedad *xGravity* se encargará de los movimientos secundarios, puede utilizar la propiedad *yGravity* para manejar los tipos de inclinación arriba / abajo.

Oyente del Acelerómetro

Los acontecimientos del Acelerómetro están basados en la ejecución temporal, así que usamos la palabra clave *Runtime* para añadir el oyente.

```
Runtime:addEventListener("accelerometer", acc)
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

4. DESARROLLAR UNA APLICACIÓN DE ENTRETENIMIENTO ‘’MAGIC BALL’’ (LA PELOTA MÁGICA)

Utilizando el *Shake event* construido en el **Corona API**, vamos a crear una aplicación que genera un resultado aleatorio de palabras predefinidas. Tú también aprenderás a crear animaciones sencillas utilizando los métodos de transición.

Selección del dispositivo de destino

Lo primero que hay que hacer es seleccionar la plataforma en la que se desea ejecutar su aplicación, de esta manera se podrá elegir el tamaño de las imágenes que se van a utilizar.

La plataforma **iOS** tiene las siguientes características:

- **iPad**: 1024x768px, 132 ppi
- **iPhone/iPodTouch**: 320x480px, 163 ppi
- **iPhone 4**: 960x640px, 326 ppi

Para **Android**, es un poco diferente, al ser una plataforma abierta, usted puede encontrar muchas resoluciones de pantalla diferentes:

- **Nexus One**: 480x800px, 254 ppi
- **Droid**: 854x480px, 265 ppi
- **HTC Legend**: 320x480px, 180 ppi

Interfaz

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Figura 8: Magic Ball

Esta es la gráfica interfaz que va a utilizar, se incluye un gráfico triangular que servirá de Octohedron encontrado en Magic Balls.

Código

En primer lugar, ocultar la barra de estado, esto es la barra en la parte superior de la pantalla que muestra los indicadores de la hora, de la señal y otros.

```
display.setStatusBar(display.HiddenStatusBar)
```

Fondo

Ahora añadimos el fondo app.

```
local background = display.newImage("background.png")
```

Esta línea crea el fondo variable local y utiliza la API de pantalla para añadir la imagen indicada en el escenario. Por defecto, la imagen se añade a 0,0 con la esquina superior izquierda como punto de referencia.

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

Octaedro

Se repite el proceso con la imagen del octaedro, colocándolo en el centro del escenario.

```
local octahedron = display.newImage("octahedron.png", 110, 186)
octahedron.isVisible = false
```

El Octaedro será invisible por defecto, y aparecerá en la sacudida del primer dispositivo.

TextField

El siguiente código crea el centro TextField centro que mostrará la frase aleatoria cuando el evento vibratorio se envía.

```
local textfield = display.newText("", 0, 0, native.systemFontBold, 14)

textfield:setReferencePoint(display.CenterReferencePoint)
textfield.x = display.contentWidth * 0.5
textfield.y = display.contentHeight * 0.5
textfield:setTextColor(255, 255, 255)
```

Variables necesarias

Las siguientes variables se utilizan para controlar la vibración.

- **Vibrar**: Una tabla que se utilizará como función de escucha para el evento vibratorio.
- **Opciones**: Se guardan las palabras que se puedan mostrar por la bola mágica.

```
local shake = {}
local options = {"Probably Not", "No.", "Nope", "Maybe", "Yes", "Probably", "It's
Done", "Of Course"}
```

Función vibratoria

Esta función detecta una vibración y muestra el octoedro y el texto si es cierto.

```
function shake:accelerometer(e)
	if(e.isShake == true) then
		octohedron.isVisible = true
		transition.from(octohedron, {alpha = 0}) -- Show octohedron
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

```
            textfield.text = options[math.random(1, 8)]
            transition.from(textfield, {alpha = 0})
        end
end
```

Escucha del acelerómetro

Los acontecimientos del acelerómetro están basados en un tiempo de ejecución, así que usamos la palabra clave *Runtime* para añadir al oyente.

```
Runtime:addEventListener("accelerometer", shake)
```

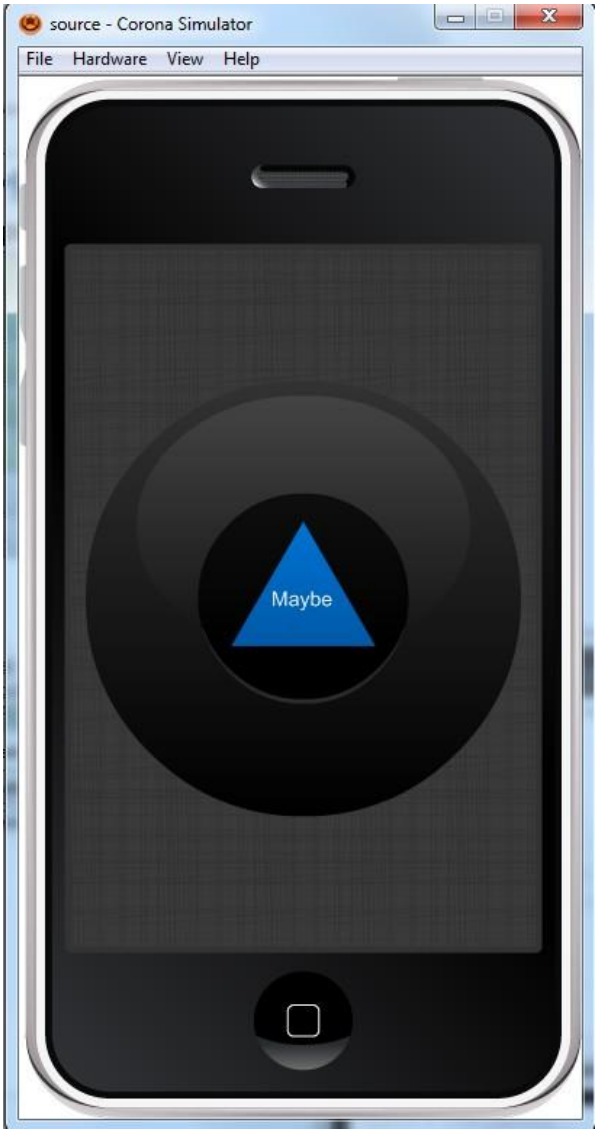


Figura 9: Magic Ball - Resultado final

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

5. TRABAJAR CON ALERTAS

Las alertas son un método predefinido del sistema para mostrar la información al usuario, comúnmente se utilizan para mostrar mensajes cortos y pueden incluir una o múltiples opciones para determinar una acción posterior.

Selección del dispositivo de destino

Lo primero que tienes que hacer es seleccionar la plataforma en la que desea ejecutar su aplicación, de esta manera se podrá elegir el tamaño de las imágenes que se van a utilizar.

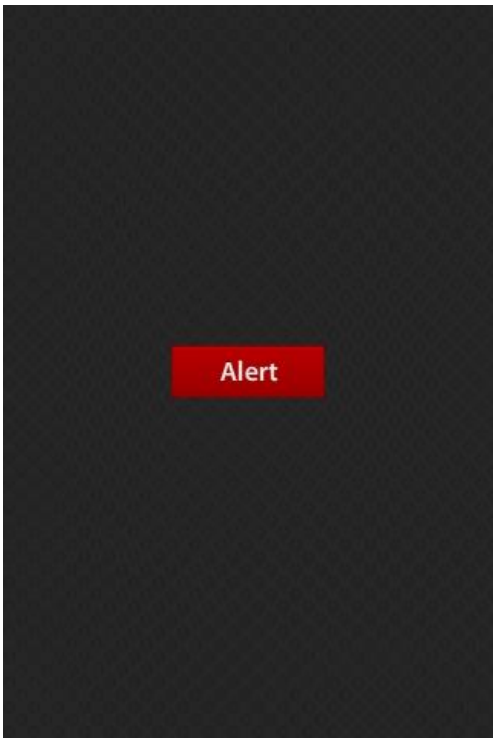
La plataforma iOS tiene las siguientes características:

- **iPad**: 1024x768px, 132 ppi
- **iPhone/iPodTouch**: 320x480px, 163 ppi
- **iPhone 4**: 960x640px, 326 ppi

Para **Android**, es un poco diferente, al ser una plataforma abierta, usted puede encontrar muchas resoluciones diferentes de pantalla:

- **Nexus One**: 480x800px, 254 ppi
- **Droid**: 854x480px, 265 ppi
- **HTC Legend**: 320x480px, 180 ppi

Interfaz



Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Código

Ocultar barra de estado

En primer lugar, ocultamos la barra de estado, esto es la barra en la parte superior de la pantalla que muestra los indicadores de la hora, de la señal y otros indicadores.

```
display.setStatusBar(display.HiddenStatusBar)
```

Fondo

Esta línea crea el fondo variable local y utiliza la API de pantalla para añadir la imagen especificada en el escenario. Por defecto, la imagen se añade a 0,0 utilizando la esquina superior izquierda como punto de referencia.

```
local background = display.newImage("background.png")
```

Botón de alerta

Repetir el proceso con la imagen del botón, colocándolo en el centro del escenario. La función del botón se creará más adelante en el código.

```
local alertButton = display.newImage("alertButton.png")

alertButton:setReferencePoint(display.CenterReferencePoint)
alertButton.x = 160
alertButton.y = 240
```

Listador de Clicks de alerta

Cuando el usuario hace clic sobre cualquiera de los botones de opción en la alerta, se produce un clic, tenemos que comprobar si el índice del botón pulsado, para saber qué opción se ha seleccionado. Una alerta permite incluir hasta 6 botones, su índice se define por el orden en que fue escrito en la llamada de alerta.

```
local function onClick(e)
	if e.action == "clicked" then
		if e.index == 1 then
			-- //Some Action
			elseif e.index == 2 then
				system.openURL( "http://www.ebusiness-lab.gr" )
		end
	end
end
```

Mostrar alerta

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Esta función se ejecuta cuando el botón de alerta se pulsa, se utiliza el método de *native.showAlert ()* para mostrar la alerta. La alerta se vinculará a una variable que servirá como el ID de alerta, de esta manera puede ser localizado, reutilizado o eliminado por el método *native.cancelAlert ()*.

```
function alertButton:tap(e)
	local alert = native.showAlert("TEI Messolonghi", "Mobile Development at TEI Mesolonghi", {"OK", "Learn More"}, onClick)
end
```

Este método tiene cuatro parámetros, vamos a echarles un vistazo:

native.showAlert (**título, mensaje, botones, oyente**)

- **Título**: El texto en la parte superior de la alerta.
- **Mensaje**: El cuerpo de la alerta.
- **Botones**: Una tabla que contiene los botones que se mostrarán en la alerta, puede mostrar hasta 6 botones.
- **Oyente**: Una función que va a escuchar eventos de clic en el botón de alerta.

Escucha del botón de alerta

El botón tiene ahora una función que se ejecuta cuando se pulsa, pero esta función por sí sola no será capaz de reaccionar sin un oyente.

La siguiente línea de código establece la escucha:

```
alertButton:addEventListener("tap", alertButton)
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Figura 10: Mensaje de alerta – Simulación

6. CREAR UN JUEGO DE BALONCESTO SIMPLE CON CORONA

El motor de física que viene con la edición Juego Corona es una herramienta increíblemente poderosa y fácil de usar. En este tutorial, vamos a cubrir la realización de un partido de baloncesto rudimentario utilizando esta emocionante tecnología.

Figura 11: El final del proyecto, que se ejecuta en el simulador de iPhone

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Paso 1: Configuración del motor de física

```
1. display.setStatusBar(display.HiddenStatusBar)
2. local physics = require "physics"
3. physics.start()
4. physics.setGravity(9.81, 0)  -- 9.81 m/s*s in the positive x direction
5. physics.setScale(80)  -- 80 pixels per meter
6. physics.setDrawMode("normal")
```

Lo primero que hacemos (como en muchos programas) es deshacernos de la barra de estado situada en la parte superior de la pantalla. A continuación, hacemos lo necesario para realizar las declaraciones que se requieren para usar la física y almacenar el resultado en lo que llamamos variable "física". Las cosas se ponen más interesantes en las siguientes líneas. En la línea cinco, establecemos la aceleración gravitacional. Por lo general, la gravedad se establece en 9,8 m / s * s en la dirección y-positiva, pero en este caso queremos que la fuerza de gravedad en la dirección x-positiva ya que la aplicación tendrá una orientación horizontal. Por otra parte, se establece la escala a 80 píxeles por metro. Este número puede variar un poco en función del tamaño de los objetos en su aplicación, y puede que tenga que jugar un rato con él para dar a su juego la sensación correcta. Si se escoge 80 px / m se quiere encajar unos 15 pies de espacio vertical en la pantalla. Sabiendo eso, es sólo una simple cuestión de conversión de unidades para obtener un valor.

Nota: Es importante tratar de relacionar todo con los objetos del mundo real en aplicaciones de la física. Cuanto más se usen las mediciones de la vida real, menor conjeturas habrán y la aplicación parecerá más realista.

Complementamos estas pocas líneas configurando el modo de dibujo a la normalidad. Esta línea hace que sea más fácil cambiar al modo de depuración posterior si tuvieramos que fijar algún comportamiento no deseado con las colisiones. Al definir este valor normal, definimos el comportamiento predeterminado y se dibujan las formas como el usuario los ve en el juego final.

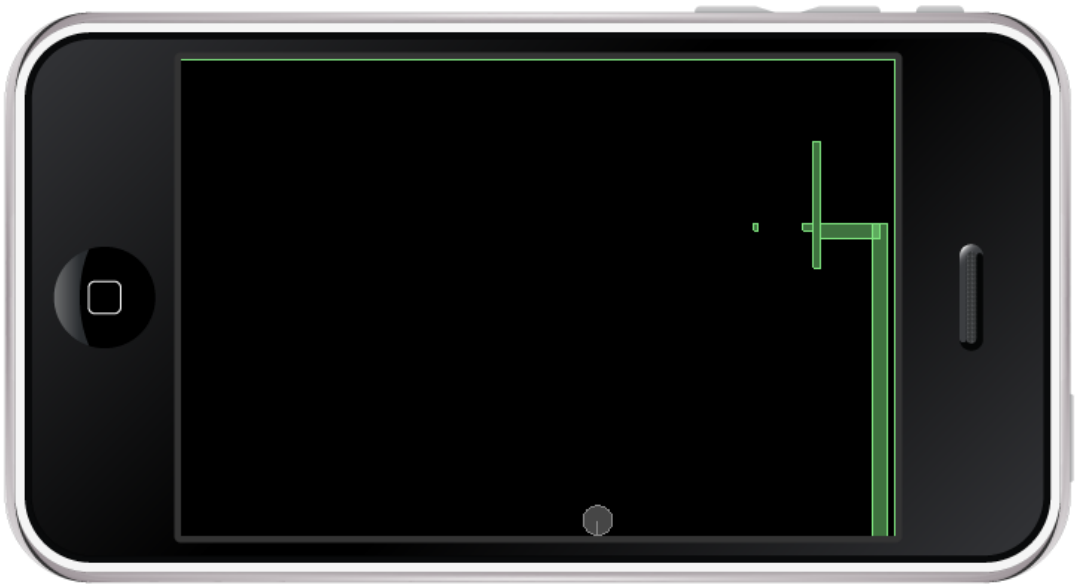


Figura 12: El proyecto final se ve en modo "debug"

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Paso 2: Creación de la Arena

```
1. local background = display.newRect(0,0,display.contentWidth,display.contentHe
   ight)
2. local score = display.newText("Score: 0", 50, 300)
3. score:setTextColor(0, 0, 0)
4. score.rotation = -90
5. score.size = 36
6. local floor = display.newRect(320, 0, 1, 480)
7. local lWall = display.newRect(0, 480, 320, 1)
8. local rWall = display.newRect(0, -1, 320, 1)
9. local ceiling = display.newRect(-1, 0, 1, 480)
10.
11. staticMaterial = {density=2, friction=.3, bounce=.4}
12. physics.addBody(floor, "static", staticMaterial)
13. physics.addBody(lWall, "static", staticMaterial)
14. physics.addBody(rWall, "static", staticMaterial)
15. physics.addBody(ceiling, "static", staticMaterial)
```

Este bloque establece los límites de la arena, y las propiedades de todos los objetos estáticos de la aplicación. Empezamos por la adición de un simple (blanco por defecto) rectángulo al fondo. En el interior del rectángulo blanco en la imagen de fondo, colocamos un poco de texto para mostrar el resultado actual. Debido a que la aplicación se muestra en el modo paisaje, también realizamos los ajustes necesarios aquí. La arena tiene que atrapar el balón dentro de la parte visible de la pantalla. Esto lo conseguimos con cuatro rectángulos estáticos (suelo, lWall, rwall, techo) colocados fuera de su vista.

A continuación, llevamos la física a la ecuación. En lugar de volver a escribir la tabla de las propiedades físicas de cada objeto, se crea un nombre de tabla *staticMaterial* para ser reutilizada para cada una de las paredes y la canasta misma. Se han elegido valores bastante estándar para estas propiedades, aunque les animo a jugar con ellos. Hay un paso más que debemos considerar, es decirle a Corona que estos objetos deben participar en los cálculos de la física. Hacemos esto llamando a la función AddBody del objeto de la física. Esta función tiene tres argumentos:

1. El objeto.
2. Un modificador opcional.
3. Una tabla de propiedades físicas.

Ya hemos determinado las propiedades y los objetos, todo lo que queda es el modificador opcional. Usamos "static" para evitar la gravedad, o cualquier otra fuerza para el caso, de desplazar nuestras paredes!

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

Figura 13: El fondo blanco, la tabla de puntuación y paredes invisibles

Paso 3: Agregar un balón y un aro

```
1.  -- Create the goal
2.  local vertPost = display.newRect(110, 5, 210, 10)
3.  vertPost:setFillColor(33, 33, 33)
4.  local horizPost = display.newRect(110, 10, 10, 40)
5.  horizPost:setFillColor(33, 33, 33)
6.  local backboard = display.newRect(55, 50, 85, 5)
7.  backboard:setFillColor(33, 33, 33)
8.
9.  physics.addBody(vertPost, "static", staticMaterial)
10. physics.addBody(horizPost, "static", staticMaterial)
11. physics.addBody(backboard, "static", staticMaterial)
12.
13. --Create the Ball
14. local ball = display.newCircle(50, 200, 10)
15. ball:setFillColor(192, 99, 55)
16.
17. physics.addBody(ball, {density=.8, friction=.3, bounce=.6, radius=10})
```

De un solo golpe, creamos el resto de los elementos visuales de nuestra aplicación. Todo esto debería ser muy familiar. Hay sólo dos cosas que me gustaría señalar. En primer lugar, algunos de los valores para el posicionamiento de la meta pueden parecer apagados. Esto es para dar cuenta de la orientación horizontal. El objetivo aparecerá en posición vertical cuando el dispositivo se gire sobre su lado. También, asegúrese de incluir la propiedad del radio en la tabla de propiedades de la pelota para que se comporte correctamente.

Figure 1: Después de añadir un balón y la canasta

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

2011-1-ES1-LEO05-35968

Paso 4: Creación de un soporte de arrastre para el balón

```
1.  local function drag( event )  -- create a function to execute when the ball
    is touched
2.      local myball = event.target    -- event.target is "who" is capturing the
    event
3.
4.      local phase = event.phase -- event.phase describes the touch sequence:
    "began", "moved", "canceled", etc...
5.      if "began" == phase then
6.          display.getCurrentStage():setFocus( myball )  -- by setting focus to
    the ball we instruct the system to deliver all future hit events to the same
    object, so that we can drag the ball around (the "moved" phase, below)
7.
8.          -- store initial position: we store the horizontal & vertical
    difference between the ball and the touch positions, so that we can drag the
    ball from anywhere on it's surface
9.          myball.x0 = event.x - myball.x
10.         myball.y0 = event.y - myball.y
11.
12.         -- avoid gravitational forces
13.         event.target.bodyType = "kinematic"
14.
15.         -- stop current motion, if any
16.         event.target:setLinearVelocity( 0, 0 )
17.         event.target.angularVelocity = 0
18.
19.     else
20.         if "moved" == phase then
21.             myball.x = event.x - myball.x0
22.             myball.y = event.y - myball.y0
23.         elseif "ended" == phase or "cancelled" == phase then
24.             display.getCurrentStage():setFocus( nil )  -- clear focus, user
    can touch any other objects after this
25.             event.target.bodyType = "dynamic"  -- re-enable gravity
26.         end
27.     end
28.
29.     return true  -- when an event handler returns true, no other handlers get
    executed after this one
30. end

myball:addEventListener("touch", drag) -- make ball listen to the touch event and
reply with the drag function
```

Esta función nos da soporte de arrastre muy básico. Algunos de los puntos culminantes incluyen el establecimiento de la carrocería de la bola a la gravedad tan cinemática para no tirar el balón fuera de las manos del usuario (Nota: ajustando esto a un movimiento dinámico después de que el toque haya terminado). Las líneas son igualmente importantes justo después. Hay que detener todo movimiento de la pelota cuando se toca para evitar el mismo problema que tuvimos con la gravedad.

Si ejecuta la aplicación como está ahora, probablemente se dará cuenta de que la pelota pierde toda su fuerza tan pronto como deje de tocarla. Para remediar esto, tendremos que crear una función para realizar un seguimiento de la velocidad de la pelota, y luego controlar la velocidad de la pelota apropiadamente después de que termine el toque.

```
1.  local speedX = 0
```

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

```
2.  local speedY = 0
3.  local prevTime = 0
4.  local prevX = 0
5.  local prevY = 0
6.
7.  function trackVelocity(event)
8.      local timePassed = event.time - prevTime  -- time is given in msec
9.      prevTime = prevTime + timePassed
10.
11.     speedX = (myball.x - prevX)/(timePassed/1000)  -- velocity is counted at
    pixels/sec
12.     speedY = (myball.y - prevY)/(timePassed/1000)
13.
14.     prevX = myball.x
15.     prevY = myball.y
16. end
17.
18. Runtime:addEventListener("enterFrame", trackVelocity)  -- trackVelocity gets
    executed everytime the screen is redrawn
```

Creamos *trackVelocity* como un detector del evento *enterFrame*, por lo que se conecta cada vez que la pantalla se redibuja. Lo que hace es encontrar el cambio en la velocidad sobre el cambio en el tiempo para encontrar la velocidad de la pelota en píxeles por segundo. Realmente no hay mucho en ello. Agregue la línea siguiente a la función de arrastre para ajustar correctamente la velocidad lineal de la pelota.

```
1.  myball:setLinearVelocity(speedX, speedY) -- when myball is released it gets
    the computed velocity
```

Paso 5: Creación del mecanismo de canasta y puntuación

Comenzamos con un trabajo más visual, pero por ahora usted debe ser un profesional en rectángulos, por lo que no debería ser costoso . El código siguiente crea el aro. Observe que la porción central del aro, no va a ser parte del sistema físico porque queremos que la bola pase libremente.

```
1.  local rimBack = display.newRect(110, 55, 5, 7)
2.  rimBack:setFillColor(207, 67, 4)
3.  local rimFront = display.newRect(110, 92, 5, 3)
4.  rimFront:setFillColor(207, 67, 4)
5.  local rimMiddle = display.newRect(110, 62, 5, 30)
6.  rimMiddle:setFillColor(207, 67, 4)
7.
8.  physics.addBody(rimBack, "static", staticMaterial)
9.  physics.addBody(rimFront, "static", staticMaterial)  -- both the back and
    front of the rim should have a static body
```

Lo siguiente que necesitamos es la manera de saber cuando el balón ha pasado a través del objetivo. La forma más sencilla de lograr esto es mediante la designación de una pequeña porción de la pantalla cerca del borde como una "zona de calificación". Cada vez que el balón esté en esta zona podemos incrementar la puntuación. Para evitar una puntuación errónea, cuando la bola permanece alrededor del borde, hacemos un seguimiento de la hora de la última canasta, y velamos por que exista una separación adecuada entre cada objetivo sucesivo. Un retraso de un segundo debería trabajarse muy bien.

```
1.  scoreCtr = 0
2.  local lastGoalTime = 1000
3.
```

Lifelong Learning Programme
Education and Culture DG
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE Organismo Autónomo Programas Educativos Europeos

```
4.  function monitorScore(event)
5.      if event.time - lastGoalTime > 1000  then  -- allow execution only after
    1 second
6.          if ball.x > 103 and ball.x < 117 and ball.y > 62 and ball.y < 92 then

7.              scoreCtr = scoreCtr + 1
8.              print(score.text)
9.              lastGoalTime = event.time
10.             score.text = "Score: " .. scoreCtr
11.         end
12.     end
13. end
14. Runtime:addEventListener("enterFrame", monitorScore)  -- scoring is monitored
    everytime the screen is redrawn
```

Conclusión

Corona se encarga de las tareas físicas más difíciles, permitiendo que pases más tiempo centrándote en el contenido y la jugabilidad del juego.

(Referencia: Carter Grove, noviembre de 2010, móviles tuts+

http://mobile.tutsplus.com/tutorials/corona/corona-sdk_game-development_basketball)

Juego de baloncesto – Código completo

Incluye archivo externo (main.lua)

Education and Culture DG
Lifelong Learning Programme
This project has been funded with support from the European Commission.
This material reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Programa de aprendizaje permanente
APEE
Organismo Autónomo Programas Educativos Europeos