KAPITEL FÜNF

PROGRAMMIEREN

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

Index

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

1. EINFÜHRUNG IN CORONA SDK: EASY CROSS-PLATFORM DEVELOPMENT

Einführung

Corona SDK ist eine exzellente Möglichkeit für jeden mobilen Entwickler, vom Anfänger bis zum Fortgeschrittenen. Diese Anleitung ist eine Einführung in eine einfach zu bedienende Plattform und zeigt, wie man beginnt, Inhalte auf der bevorzugten Plattform zu erstellen.

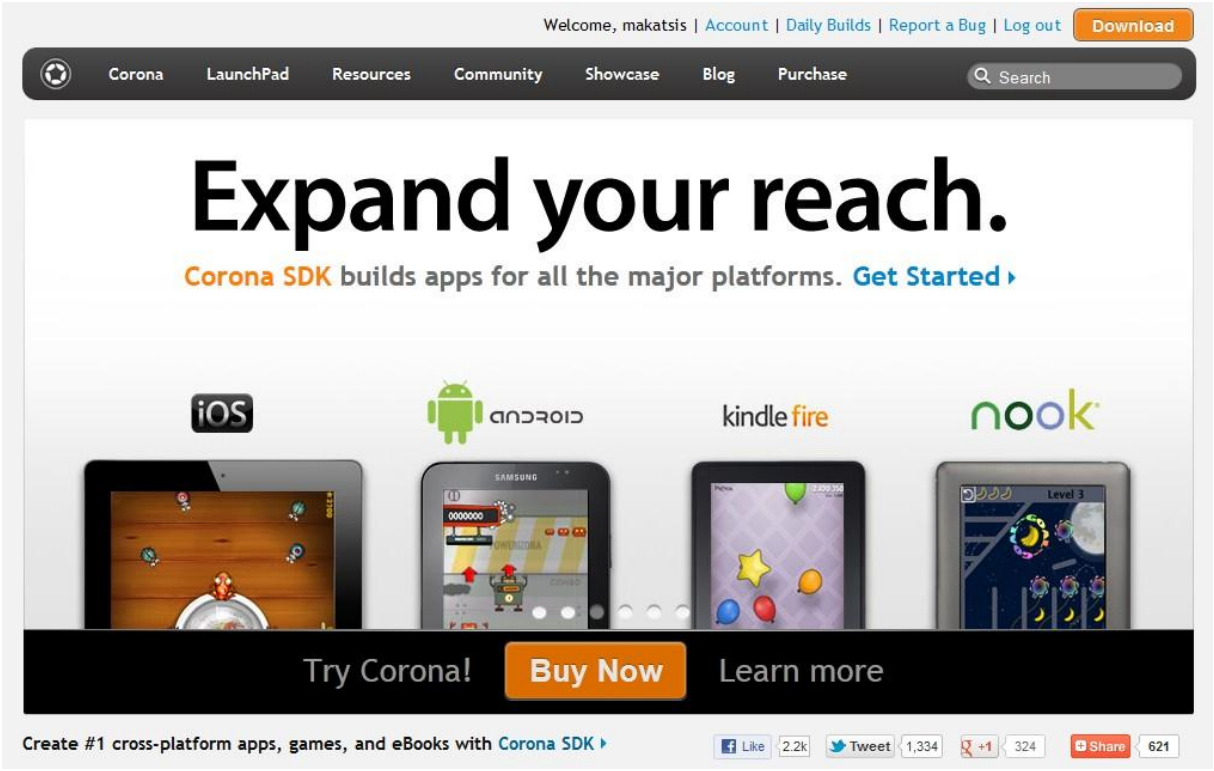


Figure 1: Official Corona Website

Die **offizielle Corona Website** beschreibt die SDK folgendermaßen:

Corona ist ein schnelles und einfaches Entwicklungstool für iPhone, iPad und Adroid Spiele und Apps.

Die Apps laufen mit 30 Bildern pro Sekunde mit nur 300k und die Grafik- und Animations-Engine nutzt vollständig die OpenGL Hardwarebeschleunigung.

Corona SDK ist das erste der Ansca's Corona Produktfamilie für das Erstellen von high-performance, multimedial anspruchsvoller Apps und Spiele für das iPhone. Mit Corona kann man Apps in wenigen Stunden erstellen. Kein C/**Cocoa** und kein **C++** erforderlich.

Corona SDK bietet viele Features, mit denen man auf eine sehr zuverlässige Art und Weise Apps erstellen kann. Einige sind:

- **Native Application Development**: Coronas ausführbare Binärdateien sind 100% **Objective-C/C++**, so dass man sich nicht um neue Entwicklungsregeln außerhalb kümmern muss. Natürlich muss Corona **Xcode** kompilieren.

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

- **Automatische OpenGL-ES Integration**: Keine Notwendigkeit umfangreiche Klassen oder Funktionen aufzurufen um einfache Manipulationen zu erstellen.
- **Cross-Platform Development**: Corona kann Apps für **iOS** (iPhone, iPod Touch, iPad) und **Android** Geräte erstellen.
- **Performance**: Corona ist für den Einsatz von hardwarebeschleunigenden Features optimiert, was eine starke Leistung in Spielen und Apps zur Folge hat.
- **Device Features**: Steuerelemente und Hardware wie Kamera, Beschleunigungsmesser, GPS, usw.
- **Einfach zu lernen**. Corona nutzt die **Lua** Programmiersprache, die sehr mächtig aber einfach zu lernen ist.

Unterstützte Plattformen

Der größte Vorteil von Corona ist, dass es ermöglicht, mit einem Basiscode zu arbeiten und Produkte für viele verschiedene Geräte zu produzieren.

Corona SDK im Speziellen ermöglicht es, Apps für **iOS**-Geräte und **Android**-Geräte zu erstellen.

Entwicklung mit Lua

Figure 2: The Lua Programming Language

Corona benutzt die **Programmiersprache** Lua, um Applikationen zu erstellen. Lua ist eine Skriptsprache, die häufig für die Entwicklung von Spielen benutzt wird.

Lua genießt eine gute Marktakzeptanz in der Entwicklercommunity. Die Lua Syntax kann mit Sprachen wie **JavaScript** oder **ActionScript 3** verglichen werden, was das Erlernen erleichtert.

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

Derzeit beschäftigen sich viele Programmiersprachen damit, wie sie helfen können, Programme mit hundertaussenden Zeilen zu schreiben.

Lua hilft Ihnen nicht dabei, Programme mit hunderttausend Zeilen zu schreiben. Stattdessen wird versucht, ein Problem mit nur hundert Zeilen oder weniger zu lösen. Um das Ziel zu erreichen, stützt sich Lua auf die Erweiterbarkeit, wie viele andere Sprachen. Im Vergleich zu anderen Programmiersprachen kann Lua nicht nur die Lua Software, sondern auch durch Software, die in anderen Sprachen wie **C** und **C++** geschrieben ist, erweitert werden.

Lua wurde von Beginn an so designt, dass Software, die in C oder anderen gebräuchlichen Sprachen geschrieben ist, integriert werden kann. Diese Dualität bringt viele Vorteile. Lua ist eine kleine, einfache Sprache, teilweise weil sie nicht etwas versucht, für das C bereits gut funktioniert, wie zum Beispiel sheer performance, low-level Operationen, oder interface mit third-party software. Lua bedient sich C für diese Aufgaben. Was Lua anbietet, ist in C nicht so gut: Distanz von der Hardware, dynamische Strukturen, keine Redundanz, einfaches Testen und Debuggen. Dafür hat Lua eine sichere Umgebung, automatisches Speichermanagement, eine großartige Möglichkeit, Strings und andere dynamische Daten handzuhaben.

Lua ist mehr als eine erweiterbare Sprache, sie ist auch eine „glue“ Sprache. Sie unterstützt einen komponentenbasierten Ansatz von Softwareentwicklung, in dem eine Applikation durch das Zusammenfügen von High-Level-Komponenten erstellt wird. Üblicherweise sind diese Komponenten in einer kompilierten, statisch geschriebenen Sprache wie C oder C++ geschrieben. Lua ist der Kleber, den wir benutzen, um diese Komponenten zusammenzusetzen und zu verbinden. Diese Komponenten (oder Objekte) repräsentieren konkretere low-level Konzepte (wie Widgets und Datenstrukturen) welche nicht vielen Änderungen während der Entwicklung unterliegen und den Großteil der the **CPU time** des fertigen Programms nehmen. Lua gibt der Applikation die Form, was möglicherweise viel während des Lebenszyklus des Produkts verändert. Daher können wir Lua nicht nur zum Zusammensetzen von Komponenten, sondern auch um sie zu adaptieren, umformen oder sogar neu zu erstellen nutzen.

Sicherlich ist Lua nicht die einzige Skriptsprache, es gibt auch andere Sprachen für mehr oder weniger die gleichen Zwecke, wie zum Beispiel **Perl**, **Tcl**, **Ruby**, **Forth**, and **Python**. Die folgenden Features unterscheiden Lua von diesen Sprachen, obwohl andere Sprache manche der Features teilen, gibt es keine Sprache mit einem gleichen Profil wie Lua:

Erweiterbarkeit: Die Erweiterbarkeit von Lua ist so auffallend, dass manche Lua nicht als Sprache, sondern als einen Baukasten für domänenspezifische Sprachen betrachten. Lua wurde von Grund auf designt, um erweitert zu werden, durch beides, Lua und externen C Code. Als ein Konzeptmerkmal implemiert es die meisten eigenen Basisfunktionen durch externe Bibliotheken. Lua lässt sich sehr einfach mit C/C++ und anderen Sprachen wie **Fortran**, **Java**, **Smalltalk**, **Ada**, und sogar mit anderen Skriptsprachen verbinden.

- **Einfachheit**: Lua ist eine einfache und kleine Sprache. Sie hat wenige (aber wirksame) Konzepte. Diese Einfachheit macht es möglich, Lua leicht zu lernen, und trägt zu einer kleinen Implementierung bei. Ihre gesamte Distribution (Quellcode, Dokumentation, und Binarität für manche Plattformen) passt auf eine Diskette.

- **Effizienz**: Lua hat eine recht effiziente Implementierung. Unabhängige Benchmarks stellen Lua als eine der schnellsten Sprachen in Bereich scripting (interpreted) languages dar.

Portabilität: Wenn wir von Portabilität sprechen, sprechen wir nicht von Lua, das auf beiden Plattformen, **Windows** und **Unix** läuft, wir reden von Lua, die auf allen Plattformen läuft: **NextStep**, **OS/2**, **PlayStation II (Sony)**, **Mac OS-9** and **OS X**, **BeOS**, **MS-DOS**, **IBM mainframes**, **EPOC**, **PalmOS**, **RISC OS**, und natürlich auf allen Arten von Unix und Windows. Der Quellcode für alle Plattformen ist virtuell der gleiche. Lua verwendet keine bedingte Kompilierung, um den Code und

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

unterschiedliche Maschinen anzupassen. Stattdessen bleibt es am Standard **ANSI (ISO) C**. Auf diese Weise muss man es nicht für eine neue Umgebung adaptieren: Wenn man einen ANSI C Compiler hat, muss man Lua nur fertig von der Stange aus kompilieren.

Ein Großteil der Stärke von Lua kommt von den Bibliotheken. Das ist kein Zufall. Einer der Hauptstärken von Lua ist die Erweiterbarkeit durch neue Typen und Funktionen. Viele Features tragen zu dieser Stärke bei. Dynamisches Schreiben ermöglicht auch zu einem hohen Grad Polymorphismus. Automatisches Speichermanagement vereinfacht das Interface, weil es nicht notwenig ist, zu entscheiden, wer verantwortlich für high-order und für das Zuordnen und Freigeben von Speicher ist oder mit Überlauf umgangen wird. High-order und anonyme Funktionen erlauben einen hohen Grad an Parameterisierung, sodass Funktionen vielseitiger werden.

Lua hat eine kleine Reihe von Standardbibliotheken. Wenn Lua in einer stark eingeschränkten Umgebung, wie es zum Beispiel eingebettete Prozessoren sein können, installiert wird, ist es sinnvoll, zu wählen, welche Bibliotheken gebraucht werden. Außerdem, wenn die Grenzen klar sind, ist es einfach in den Bibliothekscode hineinzugehen und eins nach dem anderen auszuwählen, welche Funktionen behalten werden. Es ist aber zu bedenken, dass Lua eher klein ist (trotz aller Standardbibliotheken) und in den meisten Systemen kann man alle Packages ohne Bedenken benutzen.

Lua Editoren

Im Moment gibt es keinen exklusiven Lua editor, aber es gibt andere großartige Editoren:

Free:

- **Eclipse**, benutzt das Lua Eclipse plugin.
- **LuaEdit**, LuaEdit ist ein IDE/Debugger/Script Editor für die Versionen 5.1 von Lua.
- **NotePad++**, ein kostenloser Quellcodeeditor, der verschiedene Programmiersprachen, inklusive Lua unterstützt.
- **TextWrangler**, ein mächtiger Allzwecktexteditor und Unix- und Serveradministrationstool.

Commercial:

- **TextMate**, Nur für Mac OS X erhältlich.
- **BBedit**, ein führender professioneller HTML und Texteditor für Macintosh.
- **Decoda**, eine professionelle Entwicklungsumgebung für das Debuggen von Luaskript in Ihren Applikationen.

Das erste Programm erstellen

Um mit Corona zu starten, beginnen wir mit der klassischen Hello World Applikation.

Öffnen Sie den bevorzugten Lua Editor und schreiben Sie den folgenden Code: `print("Hello World!").`

Erstellen sie einen neuen Projektordner mit dem Namen *HelloWorld* und speichern Sie die Datei als *main.lua*. Wir werden die App in den nächsten Schritten starten.

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

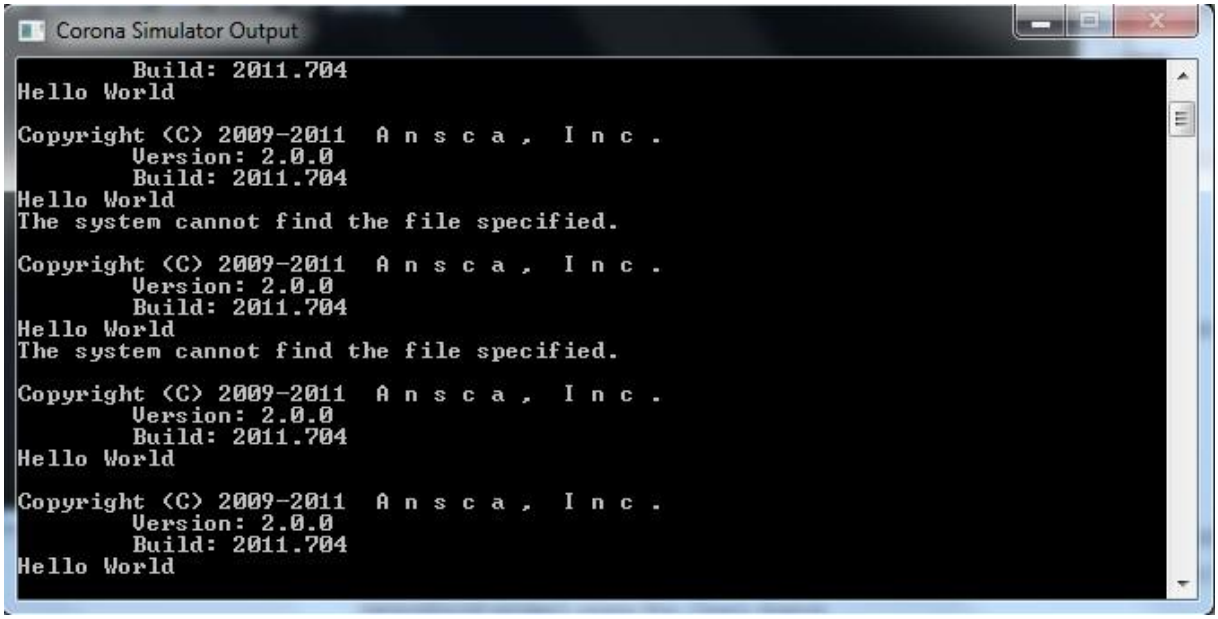


Figure 3: Corona Terminal

Es öffnet sich der *Corona Simulator,* der eine iPhone-Grafik ohne Inhalt zeigt. Dies deshalb, weil die print-Funktion nur den Terminal angibt. Um zu sehen, wie der Text im Simulator angezeigt wird, gehen wir weiter zum nächsten Schritt: Simulator

Um Zugang zum Simulation- oder Gerätebildschirm zu bekommen, müssen wir die **Corona specific API's** benutzen.

In der main.lua Datei schreiben Sie das folgendermaßen und führen das Programm erneut aus:

```
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

Figure 4: Corona Simulator

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. CORONA SDK: DIE APP ANALOGE UHR ERSTELLEN

Mit dem Corona API werden wir eine einfache analoge Uhr erstellen. Die Grafik ist ein PNG, aus einem Bildeditor Ihrer Wahl exportiert und wird dann von LUA angetrieben. Sie werden lernen, Ihre Applikation mit dem Simulator zu testen und eine App für das Gerätetesten zu erstellen.

Wählen Sie das Zielgerät

Das erste, was Sie machen müssen, ist, die Plattform zu wählen, auf der das App laufen soll. Sie können die Größe für die Bilder auswählen, die Sie benutzen werden.

Die **iOS** Plattform hat diese Charakteristiken:

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

Für **Android** ist es ein bisschen anders, da dies eine offene Plattform ist und Sie verschiedenen Bildschirmauflösungen begegnen werden:

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

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

Interface

Figure 5: Clock Interface

Code

Hintergrund

Als ersters werden wir den Hintergrund einfügen.

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

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

Diese Zeile erstellt die locale Variable *background* und benutzt das Bildschirm API, um ein bestimmtes Bild einzufügen. Standardmäßig wird das Bild bei 0,0 eingefügt.

Uhrzeiger anzeigen

Wir wiederholen den Prozess für die Uhrzeiger und das Blatt in der Mitte, indem wir sie in der Mitte des Bildschirms platzieren:

```
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)
```

Bezugspunkt

Um die Bilder richtig zu platzieren, wählen wir einen Bezugspunkt, um die Bilder relativ zu ihrer Mitte zu bewegen:

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

Ausgangsposition

Hier legen wir die Anfangsposition der Uhrzeiger fest. Wir legen nun die Rotation nach der Systemzeit fest:

```
local timeTable = os.date("*t")

hourHand.rotation = timeTable.hour * 30 + (timeTable.min * 0.5)
minuteHand.rotation = timeTable.min * 6
secondHand.rotation = timeTable.sec * 6
```

Speichergebrauch

Die *timeTable* Variable brauchen wir nur einmal beim Start der Anwendung, daher ist es nicht notwendig, sie zu speichern. Um den Speicher der Variable (die eigentlich nicht viel benötigt, aber man MUSS vertraut werden mit dem Freigeben von nichtgebrauchten Variablen oder Objekten) zu löschen, setzen wir den Wert auf *nil*, auf diese Weise kümmert sich die Garbage Collection um sie:

```
timeTable = nil
```

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

MoveHands Funktion

Die nächsten Zeilen des Codes steuern die Drehung der Uhrzeiger. Es ist der gleiche Code wie zuvor, hier wurde er nur in eine Funktion eingehüllt, durch die jede Sekunde von einem *Timer* ausgeführt wird.

```
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
```

Timer

Der *Timer* it führt die angegebene Funktion jede Sekunde aus, hier die *moveHands* Funktionen, die wir im letzten Schritt erstellt haben. Die Zeit, in der er ausgeführt wird, ist durch einen dritten Parameter festgelegt, 0 ist unendlich.

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

Figure 6: Corona Simulator analogue Clock

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

3. BESCHLEUNIGUNGSMESSER APP

Mit dem **Corona API** werden wir eine Basisapplikation erstellen, die die Bewegungen des Geräts basierend auf dem Wert des Beschleunigungsmessers registriert und ein Objekt am Bildschirm bewegt.

Wählen Sie das Zielgerät

Das erste, was Sie machen müssen, ist, die Plattform zu wählen, auf der das App laufen soll. Sie können die Größe für die Bilder auswählen, die Sie benutzen werden.

Die **iOS** Plattform hat diese Charakteristiken:

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

Für **Android** ist es ein bisschen anders, da es eine offene Plattform ist und Sie verschiedenen Bildschirmauflösungen begegnen werden:

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

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

Interface

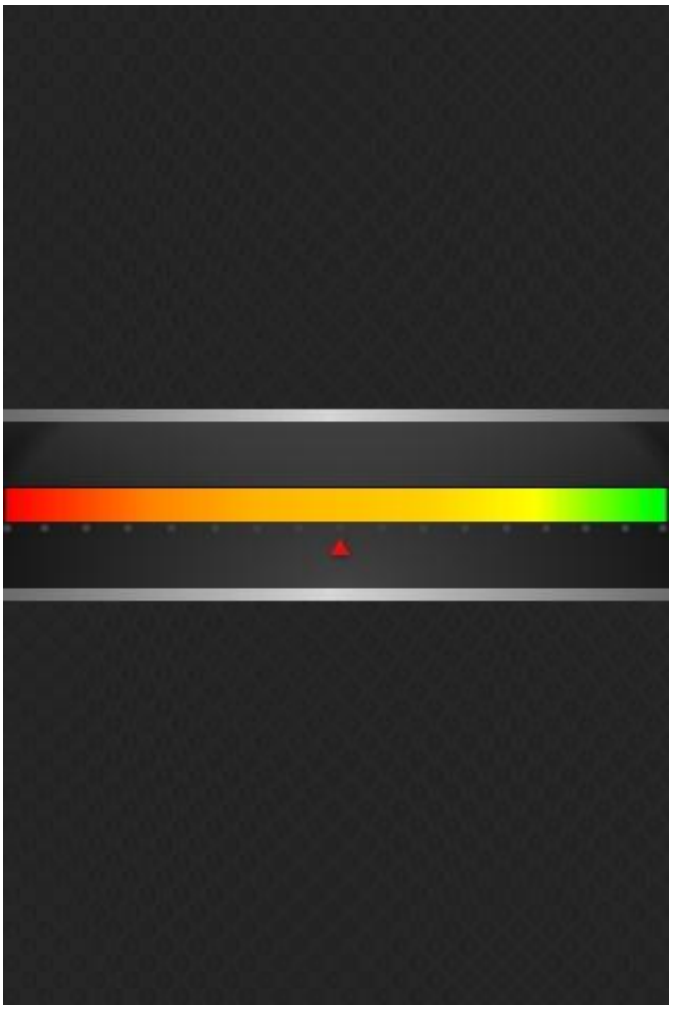


Figure 7: Accelerometer

Das ist das Grafikinterface, das wir benutzen werden. Es beinhaltet ein Dreicek, das als Positionsmesser fungiert.

Code

Statusbalken verbergen

Zuerst werden wir den Statusbalken verstecken, den Balken oben am Bildschirm, der Zeit, Signal und andere Indikatoren anzeigt.

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

Hintergrund

Nun fügen wir die Applikation background hinzu.

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.

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

Diese Zeile erstellt die locale Variable *background* und benutzt das Bildschirm API, um ein bestimmtes Bild einzufügen. Standardmäßig wird das Bild bei 0,0 eingefügt und die linkere obere Ecke als Bezugspunkt herangezogen.

Indicator

Wir wiederholen den Vorgang mit der Position des Indikatorbildes, das wir in der Mitte des Bildschirms platzieren.

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

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

Variablen

Die weiteren Variablen, die benutzt werden, um das accelerometer Ereignis zu bedienen:

- **acc**: Eine Tabelle, die als Funktion listener für den Beschleunigungsmesser benutzt wird
- **centerX**: Speichert die horizontalen Werte des Bildschirms.

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

Accelerometer Funktion

Diese Funktion benutzt die *acc* Tabelle um einen Listener für den Beschleunigungsmesser zu erstellen, die *xGravity* Eigenschaft (Teil des Beschleunigungsmessers) und die centerX Variable bewegen den Positionsindikator entsprechend der berechneten Position.

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

Das balanciert unseren Indikator, wenn sich die Geräteneigung verändert, die *xGravity* Eigenschaft wird die seitlichen Bewegungen steuern, und die *yGravity* Eigenschaft die oben/unten Neigung.

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

Accelerometer Listener

Der Beschleuningungsmesser basiert auf einer runtime, also benutzen wir Runtime als Schlüsselwort für den Listener.

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

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.

4. EINE UNTERHALTSAME MAGIC BALL APP ERSTELLEN

Mit Hilfe des *Shake* Ereignisses von **Corona API** werden wir eine Applikation erstellen, die zufällige Ergebnisse von vordefinierten Worten generiert. Man lernt außerdem, einfache Animationen zu erstellen mit den *transition* Methoden.

Wählen Sie das Zielgerät

Das erste, was Sie machen müssen, ist, die Plattform zu wählen, auf der das App laufen soll. Sie können die Größe für die Bilder auswählen, die Sie benutzen werden.

Die **iOS** Plattform hat diese Charakteristiken:

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

Für **Android** ist es ein bisschen anders, da es eine offene Plattform ist und Sie verschiedenen Bildschirmauflösungen begegnen werden:

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

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

Interface

Figure 8: Magic Ball

Das ist das Grafikinterface, das wir benutzen werden. Es beinhaltet ein Dreieck, das als Oktaeder in Magic Balls dient.

Code

Zuerst werden wir den Statusbalken verstecken, den Balken oben am Bildschirm, der Zeit, Signal und andere Indikatoren anzeigt.

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

Background

Nun fügen wir die Applikation background hinzu.

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

Diese Zeile erstellt die locale Variable *background* und benutzt das Bildschirm API, um ein bestimmtes Bild einzufügen. Standardmäßig wird das Bild bei 0,0 eingefügt und die linkere obere Ecke als Bezugspunkt herangezogen.

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

Octohedron

Wir wiederholen den Vorgang mit dem Oktaeder Bild, das wir in der Mitte des Bildschirms platzieren.

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

Es ist stanardmäßig nicht sichtbar und wir erste beim ersten Schütteln des Gerätes erscheinen.

TextField

Der folgende Code erstellt das im Zentrum liegende TextField, das einen zufälligen Satz anzeigt, wenn ein Shake Ereignis ausgeführt wird.

```
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)
```

Variablen

Die nächsten Variablen, die benutzt werden um das Shake Ereignis zu steuern.

- **shake**: Eine Tabelle, die als Funktion listener für das Shake Ereignis benutzt wird.
- **options**: Speichert die Wörter, die im magischen Ball erscheinen können.

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

Shake Funktion

Diese Funktion sucht nach einem ShakeEreignis und zeigt das Oktaeder und den Text, wenn es wahr (true) ist.

```
function shake:accelerometer(e)
	if(e.isShake == true) then
		octohedron.isVisible = true
		transition.from(octohedron, {alpha = 0}) -- Show octohedron
		textfield.text = options[math.random(1, 8)]
		transition.from(textfield, {alpha = 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

```
        end
end
```

Accelerometer Listener

Die Accelerometer Ereignisse basieren auf einer runtime, also benutzen wir Runtime als Schlüsselwort um den Listener hinzuzufügen.

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

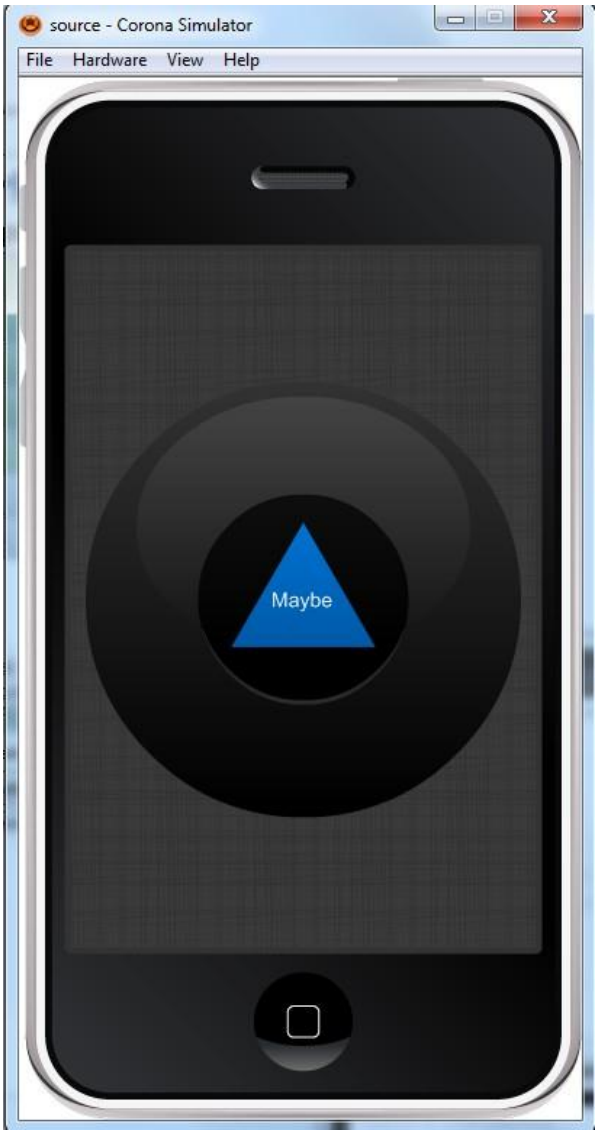


Figure 9: Magic Ball - Final Result

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.

5. ARBEITEN MIT ALERTS

Alerts sind eine vordefinierte Systemmehtode, um den Usern Informationen zu zeigen. Sie werden üblicherweise für kurze Nachrichten eingesetzt und können eine oder mehrere Optionen beinhalten, um eine spätere Aktion zu ermitteln.

Wählen Sie das Zielgerät

Das erste, was Sie machen müssen, ist die Plattform zu wählen, auf der das App laufen soll. Sie können die Größe für die Bilder auswählen, die Sie benutzen werden.

Die **iOS** Plattform hat diese Charakteristiken:

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

Für **Android** ist es ein bisschen anders, da es eine offene Plattform ist und Sie verschiedenen Bildschirmauflösungen begegnen werden:

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

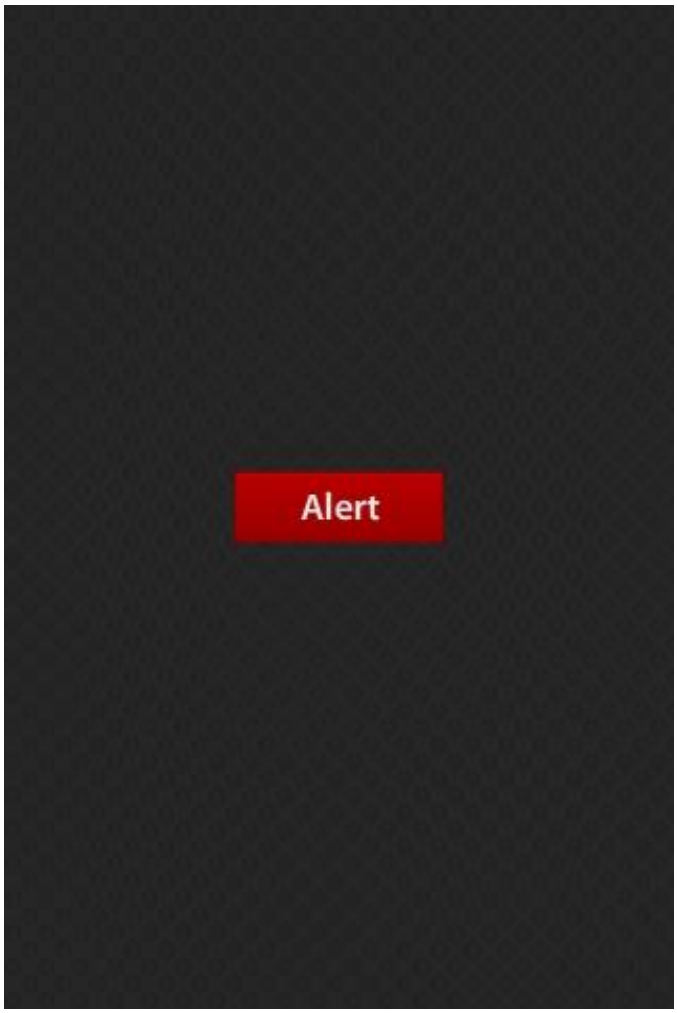
Education and Culture DG
Lifelong Learning Programme

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

Interface



Code

Hide Status Bar

Zuerst werden wir den Statusbalken verstecken, den Balken oben am Bildschirm, der Zeit, Signal und andere Indikatoren anzeigt.

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

Background

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

Diese Zeile erstellt die locale Variable *background* und benutzt das Bildschirm API, um ein bestimmtes Bild einzufügen. Standardmäßig wird das Bild bei 0,0 eingefügt und die linkere obere Ecke als Bezugspunkt herangezogen.

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

Alert Button

Wir wiederholen den Prozess mit dem Button Bild, das wir in der Mitte des Bildschirms platzieren. Die Button Funktion wird später im Code erstellt.

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

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

Lister für Alert Clicks

Wenn der User auf eine der Optionsschaltflächen klickt, wird im Alert ein *clicked* Ereignis durchgeführt. Wir müssen im *index* der gedrückten Schaltfläche prüfen, welche Option ausgewählt wurde. Ein Alert kann bis zu 6 Schaltflächen erhalten, der Index wird durch die Reihenfolge bestimmt.

```
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
```

Display Alert

Diese Funktion wird ausgeführt, wenn die alert Schaltfläche gedrückt wird, und benutzt die Methode *native.showAlert(),* um den alert anzuzeigen. Der alert wird zu einer Variable verbunden, die als alert ID dient. Auf diese Art kann sie lokalisiert, wiederbenutzt oder mit der Methode *native.cancelAlert()* entfernt werden.

```
function alertButton:tap(e)
```

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

```
      local alert = native.showAlert("TEI Messolonghi", "Mobile Development at TEI
Mesolonghi", {"OK", "Learn More"}, onClick)

end
```

Die Methode hat vier Parameter:

native.showAlert(**title**, **message**, {**buttons**}, **listener**)

- **title**: Der Text oben auf dem alert.
- **message**: Der Hauptteil des alerts.
- **buttons**: Eine Tabelle, die die Schaltfächen, die angezeigt werden, beinhaltet, man kann 6 Schaltfllächen anzeigen.
- **listener**: Eine Funktionen, die nach Klickereignissen sucht.

Alert Button Listener

Die Schaltfläche hat nun eine Funktion, die gestartet wird, wenn die Schaltfläche gedrückt wird, aber diese Funktion alleine kann nicht ohne Listener reagieren.

Die nächste Zeile des Coades legt einen Listener fest.

```
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

Figure 10: Alert message – Simulation

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

6. Ein einfaches Basketball Spiel mit Corona erstellen

Die physikalische Engine der Corona Game Edition ist ein Tool, das sehr viel kann und einfach zu bedienen ist. In dieser Anleitung geht es um die Erstellung eines Basketballspiels.

Figure 11: The final project, running on the iPhone simulator

Schritt 1: Die Physics Engine erstellen

```
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")
```

Das erste, was wir machen, ist es, den Statusbalken am oberen Bildschirmrand auszublenden. Danach treffen wir notwendige Anforderungen und speichern das Ergebnis in der Variable “physics”.

In Zeile fünf legen wir die Fallbeschleunigung fest. Normalerweise wird sie auf 9,8 m/s*s in der positiven Y-Richtung festgelegt, aber hier wollten wir stattdessen die Schwerkraft auf die positive X-Richtung anwenden, da wir im Querformat sind. Dann setzen wir den Maßstab auf 80 Pixel pro Meter. Diese Zahl kann ein wenig variieren, um dem Spiel ein gutes Feeling zu geben.

Hinweis: Bei Apps mit Physik ist es wichtig, alles mit real world Objekten zu versuchen und zu erproben. Je mehr reale Maße sie benutzen, desto weniger ist es ein Rätselraten und desto realistischer wird Ihre App sein.

Wir runden diese wenigen Zeilen ab, indem wir draw mode auf normal setzen. Das macht es einfacher, den Debug Modus später zu verändern, wenn wir ungewolltes Verhalten mit Kollisionenen beheben möchten. Das auf normal zu setzen, ist das Standardverhalten und zeichnet die Formen so, wie man es im fertigen Spiel sehen wird.

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

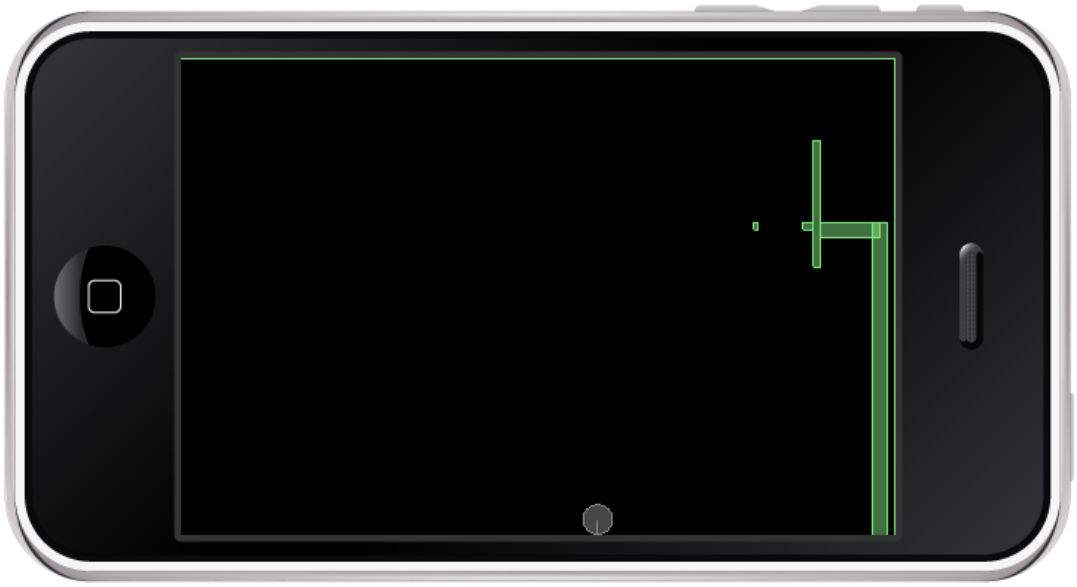


Figure 12: The final project viewed in "debug" mode

Schritt 2: Die Arena erstellen

```
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)
```

Dieser Block legt die Grenzen der Arena und die Eigenschaften aller statischen Objekte fest. Wir beginnen mit dem Hinzufügen eines einfachen (standardmäßig weiß) Rechtecks zum Hintergrund. Darin platzieren wir einen Text, um den aktuellen Spielstand anzuzeigen. Weil die App im Querformat angezeigt wird, müssen wir auch die notwendige Anpassung der Drehung machen. Die Arena muss innherhalb des sichtbaren Bildschirmteils liegen. Wir erreichen das mit vier statischen Rechtecken (floor, lWall, rWall, ceiling), die außerhalb des Sichtfelds platziert werden.

Danach bringen wir die Physik wieder ins Gleichgewicht. Anstatt die Tabelle für die physikalischen Eigenschaften jedes Objekts noch einmal zu tippen, erstellen wir eine Tabelle mit dem Namen staticMaterial, die für jede der Wände und das Ziel benutzt wird. Ich habe meist Standardwerte für die Eigenschaften genommen, obwohl man mit ihnen herumspielen kann. Es gibt noch einen Schritt, den wir gehen müssen, und das ist, Corona zu sagen, dass diese Objekte in den physikalischen Berechnungen teilhaben sollen. Wir machen das mit der addBody Funktion des Objekts physics. Diese Funktion hat drei Argumente:

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

1. Das Object
2. Einen optionalen Modifikator
3. Eine Tabelle der physikalischen Eigenschaften

Wir haben bereits die Eigenschaften und die Objekte bestimmt, so bleibt nur mehr der optionale Modifikator. Wir wählen „static“, um die Schwerkraft oder einer anderen Kraft, die eine Rolle spielt, davon abzuhalten, unsere Wände zu verschieben.

Figure 13: The white background, the score table & invisible walls

Schritt 3: Einen Ball und ein Ziel hinzufügen

```
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})
```

Auf einen Schlag werden die restlichen visuellen Elemente des Apps erstellt. Es gibt zwei wichtige Hinweise. Erstens können manche der Werte für die Position des Ziels abseits scheinen. Das ist für das Querformat. Das Ziel erscheint aufrecht, wenn das Gerät zur Seite gedreht wird. Stellen Sie auch sicher, dass die Radius Eigenschaft in der Tabelle „Eigenschaft des Balls“ beinhaltet ist, so dass er sich immer richtig verhält.

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

Figure 14: After adding a ball and a goal

Schritt 4: Drag Support für den Ball erstellen

```
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
```

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

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

Diese Funktion gibt uns eine sehr einfach Drag Unterstützung. Einige der Höhepunkte beinhalten das Festlegen des bodyType zu kinematic, so dass die Schwerkraft den Ball nicht aus den Händen fallen lässt (**Hinweis: Achten Sie darauf, dass dies auf dynamic zurückgesetzt wird, wenn es keine Berührung mehr gibt**). Die Zeilen danach sind gleich wichtig. Hier stoppen wir die Bewegung des Balls, wenn er berührt wird, um das gleiche Problem wie bei der Schwerkraft zu verhindern.

Wenn sie die App nun starten, werden Sie vielleicht feststellen, dass der Ball seinen Schwung verliert, wenn man ihn angreift. Um Abhilfe zu schaffen, müssen wir eine Funktion erstellen, die die Geschwindigkeit des Balls erfasst und diese entsprechend festlegt, nachdem die Berührung endet.

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

Wir erstellen trackVelocity als Listener für das enterFrame Ereignis, sodass es ausgeführt wird, wenn der Bildschirm neu aufgebaut wird. Es findet die Veränderungen in der Geschwindigkeit über die Veränderungen der Zeit, um die Schnelligkeit des Balls in Pixel per Sekunde zu bekommen. Fügen Sie die folgende Zeile in die drag Funktion, um die lineare Geschwindikeit des Balls richtig festzulegen.

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

Schritt 5: Den Basketballkorb und den Mechanismus für den Punktestand erstellen

Der folgende Code erstellt den Rand des Korbs. Beachten Sie, dass der mittlere Teil nicht Teil des physikalischen Systems sein wird, weil der Ball frei durchgehen soll.

```
1. local rimBack = display.newRect(110, 55, 5, 7)
2. rimBack:setFillColor(207, 67, 4)
```

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

```
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
```

Wir brauchen etwas, um zu wissen, wann der Ball durch das Ziel gegangen ist. Der leichteste Weg, das zu erreichen, ist die Bennenung einens kleinen Bereichs auf dem Bildschirm als « score zone » in der Nähe des Randes. Immer, wenn der Ball in dieser Zone ist, können wir den Spielstand erhöhen. Um eine falsche Wertung zu verhinden, wenn der Ball am Rand bleibt, merken wir uns die Zeit des letzten Treffers und versichern, dass es eine angemessene Pause zwischen den erfolgreichen Treffern gibt. Eine Sekunde sollte gut funktionieren.

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

Conclusion

Corona kümmert sich um die schwereren physikalischen Aufgaben und lässt mehr Zeit, um sich auf den Inhalt und die Spielsituation zu konzentrieren.

(Verweis: Carter Grove, November 2010, mobile tuts+ http://mobile.tutsplus.com/tutorials/corona/corona-sdk_game-development_basketball)

Basketball Spiel - Voller Code

Included external file (main.lua)

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.