Prototipo de juego 2D multiplataforma para móviles

Cross-platform 2D game prototype for mobile devices

David Hernández Bethencourt
Departamento de Ingeniería Informática
Escuela Técnica Superior de Ingeniería Informática

Trabajo de Fin de Grado

La Laguna, 7 de septiembre de 2014

Dña. **Isabel Sánchez Berriel**, con N.I.F. 42.885.838-S profesora colaboradora de Universidad adscrita al Departamento de Ingeniería Informática de la Universidad de La Laguna

C E R T I F I C A

Que la presente memoria titulada:

"Prototipo de Juego 2D multiplataforma para móviles."

ha sido realizada bajo su dirección por D. David Hernández Bethencourt, con N.I.F. 79.061.973-W.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 6 de septiembre de 2014

Agradecimientos

En primer lugar me gustaría mostrar mi profundo agradecimiento a la directora de este Trabajo de Fin de Grado, Dña. Isabel Sánchez Berriel, quien me ha guiado durante la elaboración de este proyecto para poder lograr los objetivos planteados satisfactoriamente.
Gracias por toda la paciencia, ha sido una experiencia muy agradable y enriquecedora.

En segundo lugar, me gustaría agradecerles a mi familia, a mi novia y a mis amigos, por todo el apoyo que he recibido por su parte, dado que la finalización de este trabajo supone el fin de una etapa muy importante en mi vida, de la cuál ellos han formado parte.

Finalmente, me gustaría agradecer a la ETSII, por todas las vivencias que he tenido allí y que me han formado tanto profesionalmente como personalmente.

Resumen

El objetivo de este trabajo ha sido investigar y estudiar por una parte, el arte del desarrollo de juegos 2D mediante tecnologías web HTML5 y JavaScript, y por otra parte, investigar y estudiar las herramientas actuales para el desarrollo multiplataforma y cómo convertir los juegos desarrollados en aplicaciones para dichas plataformas.

Como fruto de ese estudio, se ha llevado a cabo un prototipo de juego de puzzles 2D, denominado Puzzleyap, dirigido a Android e iOS.

Una vez definidos los requisitos del juego Puzzleyap, se ha realizado un estudio previo tanto de las plataformas existentes, como de diversos juegos que cumplan con los mismos requisitos que los propuestos para el prototipo.

Finalmente, se ha tomado la decisión de emplear diversas tecnologías para llevar a cabo el desempeño de nuestro prototipo en función de dicha investigación, obteniendo como resultado una aplicación móvil funcional que nos permite jugar un juego de puzzles 2D tanto en iOS como en Android

Palabras clave

Aplicación, móvil, PhoneGap, Puzzleyap, JavaScript, HTML5, CSS, Android, iOS, CocoonJS, juego, puzzle, 2D, multiplataforma, Appcelerator Titanium, Corona SDK.

Abstract

The objective of this work was to investigate and study on the one hand, the 2D game development's art using HTML5 and JavaScript web technologies, and secondly, to investigate and study the current tools for cross-platform development and how to convert games developed in applications for these platforms.
As a result of this study, it has been built a 2D puzzle game prototype called Puzzleyap, targeting Android and iOS.
After defining the Puzzleyap set requirements, there has been a previous study of both existing platforms, and various games that meet the same requirements as those proposed for the prototype.
Finally, we have taken the decision to use various technologies to carry out the performance of our prototype based on this research, resulting in a functional mobile application that allows us to play a 2D puzzles game both iOS and Android.

Keywords

Application, mobile, PhoneGap, Puzzleyap, JavaScript, HTML5, CSS, Android, iOS, CocoonJS, game, puzzle, 2D, cross-platform, Appcelerator Titanium, Corona SDK.

Índice General

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

La industria de los Videojuegos (se estima un volumen de 100.000 millones de dólares en 2017 [1]) ha adquirido un auge tal que se posiciona por encima de otros sectores del ocio como el cine o la música. Por otra parte, el acceso a Internet desde dispositivos móviles comienza a ser superior que el acceso a Internet desde equipos de escritorio. Es evidente además que la evolución de estos dispositivos sigue un ritmo frenético respecto a capacidades y funcionalidades. La expansión de los Smartphone y tablets suponen una revolución tecnológica, que abre unos nichos de mercado que nadie pone en duda. En el panorama actual también destaca la fuerza que están adquiriendo los juegos online. Otro de los aspectos a tener en cuenta respecto a cualquier desarrollo para dispositivos móviles y tablets es la segmentación de los dispositivos, hecho que obliga a implementar la misma aplicación para cada una de las plataformas en las que se quiera tener presencia. Como solución a este inconveniente han venido tomando auge distintas herramientas que facilitan el desarrollo de aplicaciones multiplataforma, surgiendo así la dicotomía "aplicaciones nativas vs aplicaciones web". En una aplicación nativa se desarrolla utilizando algún framework para cada entorno con algún lenguaje de programación (Java para Android móviles Blackberry, Objective-C para iOs, C/C++ para Tablet Os de Blackberry, C# o Visual Basic .NET para Windows Phone). Estas aplicaciones se compilan y se instalan en el teléfono y no necesitan de un navegador para ser ejecutadas. En contraposición las aplicaciones web se ejecutan desde un navegador, que puede estar integrado en la propia aplicación. Se desarrollan utilizando las tecnologías de la web: HTML5 [2], JavaSript [3] y CSS [4].

HTML5 representa vía para el "desarrollo estándar" ofreciendo la posibilidad de implementar el mismo juego para iPhone5S, Android S4, Android S3, Mac, iPad, Galaxy Tab, Windows 8, Smart-TVs o consolas con la reducción de costes que implica el que no sean necesarios múltiples desarrollos. La última actualización de este lenguaje ha reducido la dependencia del mismo con respecto a plugins de terceras partes en materia de audio y video.
Además de todo esto, HTML5 simplifica el proceso de instalación y actualización porque se pueden integrar fácilmente con los servicios habituales del sitio web. Por el contrario, como desventaja se encuentra la pérdida de rendimiento y las dificultades que se pueden encontrar al acceder a accesorios del dispositivo como brújula, cámara, acelerómetro o GPS.

1.1 Motivación

El presente proyecto tiene por objeto principal el estudio del estado del arte del desarrollo de juegos 2D con HTML5 y JavaScript, por una parte, y, por otra, estudiar las plataformas que permiten convertir el juego desarrollado en aplicaciones para Android o iOS.
Como resultado de dicho estudio, se desarrollará un prototipo de juego de puzzles 2D empleando dichas características.

La motivación del proyecto se fundamenta en tres razones:

1. **Oportunidad de desarrollo de un juego.** La industria del videojuego es un mercado muy amplio con múltiples salidas, lo que hace de esta una oportunidad excelente para llevar a cabo un acercamiento al mismo debido a lo que supone enfrentar el reto de desarrollar el prototipo de un juego.

2. **Desarrollo multiplataforma para móviles.** En un mundo donde debemos adaptarnos continuamente, realizar un código que podamos reutilizar para las diferentes plataformas es el mejor ejemplo de adaptación. Además de poder llegar a más dispositivos móviles, algo que interesa bastante, el desarrollo multiplataforma (en este caso para Android e iOS aunque es extensible en el futuro) permite ahorrar en tiempo, costo y en mantenimiento en función de los requisitos de la aplicación.

   Aunque el código esté dirigido a aplicaciones móviles, al ser una aplicación web, podrá ejecutarse sin ningún problema desde el navegador en cualquier PC.

3. **Conversión del juego en aplicación.** Si bien es posible la creación de un juego empleando tecnologías web (HTML5 y JavaScript), una de las cuestiones más importantes es conocer de qué herramientas disponemos los desarrolladores para poder transformar dicho juego en una aplicación para móvil. Esto es conveniente ya que siempre será más fácil acceder al propio juego desde una app, que tener que estar navegando hacia la URL del juego a través del navegador manualmente.

1.2 Objetivos y requisitos

1. Investigar las posibilidades de las tecnologías web HTML5 y JavaScript para el desarrollo de juegos 2D.
2. Investigar las diferentes plataformas que permiten el desarrollo del juego y su posterior conversión en aplicación para Android o iOS.
3. Realizar un prototipo de juego de puzzles 2D empleando dichas características.

Los requisitos del juego al que denominaremos a partir de ahora "Puzzleyap" son los siguientes:

- Multiplataforma, dirigido a Android e iOS.
- Se deberá poder sacar una fotografía con la cámara del dispositivo para la construcción y realización del puzzle.
- El puzzle será de tipo "Jigsaw" [5], es decir, se resuelve arrastrando las piezas del puzzle, hasta colocarlas en la posición adecuada, y los bordes de éstas no serán rectos. Esta diferenciación es importante, debido a que existen diversos tipos de puzzle, como los de deslizamiento (Slider en inglés), en el cual se deslizan las piezas hasta formar una configuración final.
- Se estudiará la posibilidad de añadir algún efecto sobre la imagen tras completar el puzzle.
- Se estudiará la posibilidad de que el puzzle comience a resolverse cuando agites el dispositivo.
- Se estudiará la posibilidad de implantar modo multijugador en función de la disponibilidad de tiempo.

1.3 Antecedentes y trabajos previos

Dado que la segunda parte de este proyecto consiste en el desarrollo de un prototipo de juego de puzzle 2D y, que no es una idea novedosa, la posibilidad de que existan aplicaciones que cumplan nuestros requisitos tanto en la AppStore [6] como en Google Play [7] es elevada.

Por tanto, es conveniente hacer un análisis de dichas aplicaciones para contrastar si realmente es así. Aunque no tengamos forma de saber mediante qué tecnologías o herramientas se han desarrollado las aplicaciones, podemos saber si cumplen los requisitos básicos de nuestro prototipo y si están dirigidas a una única plataforma o varias.

La siguiente tabla expuesta a continuación recoge los datos de unas 15 aplicaciones en total recogidas de iTunes o en Google Play siguiendo como criterio de búsqueda las palabras clave “puzzle” y “jigsaw”:

Nota: es posible que se repita el nombre de alguna aplicación. Esto no es debido a un error, sino que existen diversas aplicaciones con el mismo nombre diferenciadas por el desarrollador.
Para más información, consultar el Anexo. Juegos de puzzles estudiados:

Aplicación	Plataforma/s	Acceso a cámara	Acceso a galería	Tipo	¿Cumple nuestros requisitos?
Fotopuzzle	Android	Si	Si	Slide	NO
Puzzle Fotos	Android	No	Si	Slide	NO
Puzzle con tu foto	Android	Si	Si	Slide	NO
Jigsaw Puzzle	Android/iOS	Si	Si	Jigsaw	SI
Rompecabezas	Android	No	Si	Jigsaw	NO
Jigsaw Puzzles	Android/iOS/PC-Mac	No	Si	Slide	NO
Rompecabezas Jigty	Android/iOS	No	Si	Jigsaw	NO
Jigsaw Puzzle Maker	Android	No	No	Jigsaw	NO
Real Jigsaw	Android	No	Si	Jigsaw	NO
Jigsaw World	Android	No	No	Jigsaw	NO
Jigsaw Puzzles	Android/iOS	No	No	Jigsaw	NO
Jigsaw Puzzle 500+	iOS	No	No	Jigsaw	NO
Rompecabezas!	Android/iOS	No	Si	Jigsaw	NO
Puzzles y Jigsaw	Android/iOS	No	Si	Jigsaw	NO
Jigsaw Puzzles Saga	Android/iOS	Si	Si	Jigsaw	SI

Tabla 1: Aplicaciones de puzzles

Tras observar la tabla, podemos sacar varias conclusiones:

- De la cantidad de aplicaciones de puzzles que hay, sólo 2 de 15 cumplen exactamente con los requisitos propuestos para nuestro prototipo.
- Cerca de la mitad de dichos juegos se encuentran disponibles para más de una plataforma, no se ha podido determinar si se trata de un desarrollo multiplataforma con tecnologías web o si, por el contrario, son aplicaciones nativas.

También cabe destacar que debido a la potencia de HTML5, multitud de juegos de puzzles son realizados en canvas y podemos encontrarlos en la red. Aquí van algunos ejemplos:

- Jigsaw Puzzle by Raymond Hill:
  http://www.raymondhill.net/puzzle-rhill/jigsawpuzzle-rhill.php
- Grrd's Puzzle:
  https://developer.mozilla.org/ms/demos/detail/grrds-puzzle
- HTML5 Image Puzzle Game:
  http://html5puzzle.appspot.com/
- Canvas Puzzle:
  http://codecanyon.net/item/canvas-puzzle/2289964

1.4 Alcance del Proyecto

El alcance comprenderá lo establecido en los objetivos y requisitos.
No obstante, al tener que elaborar un prototipo, se llevarán a cabo una serie de iteraciones en las que se han generado los siguientes entregables:

- Entregable 1 del prototipo: Renderizado de una imagen
- Entregable 2 del prototipo: Captura de una imagen con la cámara
- Entregable 3 del prototipo: Fragmentación como puzle de la imagen y resolución del mismo

- Entregable 4 del prototipo: Aplicación de efectos tras la investigación

1.5 Destinatarios

Este proyecto está destinado especialmente a desarrolladores, dada su orientación de estudio e investigación sobre la utilidad y eficacia de herramientas multiplataforma para la programación de juegos para dispositivos móviles. Estos usuarios pueden verse en la tesitura de tener que optar por el desarrollo multiplataforma si se pretende llegar a un mayor número de usuarios. Una vez has tomado la decisión se debe investigar a qué herramientas puedes acceder para ponerte a desarrollar y ese precisamente es uno de los objetivos de este proyecto.

Capítulo 2. Estado del desarrollo de juegos 2D mediante HTML5 y JavaScript

El 17 de diciembre de 2012, el World Wide Web Consortium (W3C) [8] publica la definición completa de HTML5 y las especificaciones del Canvas 2D [9], siendo este último un lienzo de mapa de bits dependiente de la resolución de pantalla que se puede utilizar para representar gráficos, imágenes de juegos o cualquier otra información de este tipo en tiempo real. Esto se traduce en que canvas es un nuevo elemento de HTML5 que nos permite dibujar gráficos empleando el lenguaje de scripting Javascript.

2.1 HTML5 se sobrepone a Flash

Desde su aparición, HTML5 ha ido evolucionando y ha causado el descenso del trono de desarrollo de videojuegos para la Web, ostentado por Adobe Flash [10].

Esto se ha visto respaldado a lo largo de los últimos años, ya que aunque la definición completa de HTML5 se publicó en 2012, se comenzó a trabajar en el standard en el 2004.

En abril de 2010 se convirtió en el tema de los principales medios de comunicación cuando Steve Jobs, CEO de la empresa Apple Inc, publicó una carta titulada “Thoughtson Flash” (pensamientos sobre flash) [11], en donde concluye que “Flash ya no es necesario para ver vídeos o consumir cualquier tipo de contenido web" y que "los nuevos estándares abiertos creados en la era móvil, como HTML5, ganarán"

También durante esas fechas y posterior ese momento, Facebook se pasa a HTML5 y deja de lado flash [12].

Finalmente en noviembre de 2011, Adobe decide retirar Flash del mercado móvil [13], reconociendo que HTML5 es «la mejor solución para crear y entregar contenido en navegadores para plataformas móviles».

2.2 Compatibilidad de HTML5 con los navegadores

Una de las razones de la expansión de HTML5, es que es estable, flexible, no requiere de plugins y unifica el desarrollo, dado que no se debe escribir diferente código en función de la plataforma.

En lo que al desarrollo de juegos 2D se refiere, este ha sufrido un gran incremento debido a este intenso crecimiento, por lo que en la siguiente imagen se muestra una comparativa del soporte del elemento canvas en los navegadores:

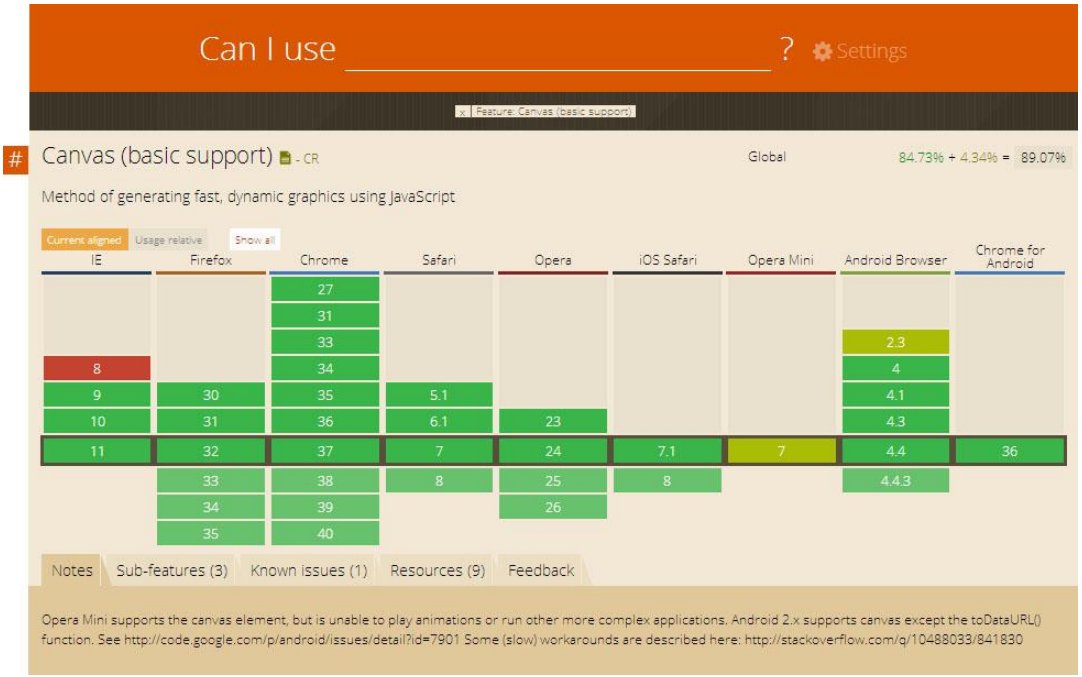

Figura 1: Compatibilidad de canvas en los navegadores [14]

Como podemos apreciar en la Figura 1, la gran compatibilidad tanto para navegadores de escritorio o de móviles hacen de HTML5 una opción más que viable para el desarrollo de juegos.

2.3 Juegos 2D y motores

Podemos encontrar infinidad de tutoriales y ejemplos en la web sobre el desarrollo de juegos 2D con HTML5. Además tenemos a disposición muchos motores 2D, frameworks y librerías que nos dan soporte para el desarrollo de nuestros juegos, llegando incluso a herramientas de creación de juegos que no requieran codificar nada, como es el caso de Construct 2 [15].

A continuación veremos cinco populares GameEngines (motores de juego) para juegos 2D. Si se desea acceder a la lista completa, se puede consultar en la siguiente web: http://html5gameengine.com/

- **Construct 2.** Construct es un editor de juego en 2D multiplataforma basado en HTML5 y desarrollado por Scirra Ltd. Está dirigida principalmente a los no programadores, permite la creación rápida de juegos en forma de arrastrar y soltar a través de un editor visual y un sistema de lógica basada en el comportamiento.

- **ImpactJS [16].** ImpactJS es un motor de juegos que permite desarrollar juegos HTML5 tanto para navegadores de escritorio como para navegadores móviles. Es muy fácil de aprender, viene con numerosos ejemplos de código, tiene una activa comunidad e incluye un robusto editor de niveles llamado Weltmeister.

- **EaselJS [17].** EaselJS proporciona soluciones directas para trabajar con gráficos ricos y la interactividad con HTML5 Canvas. Proporciona una API que es familiar a los desarrolladores de Flash, pero que abarca las sensibilidades de Javascript. Se compone de una lista completa jerárquica pantalla, un modelo de interacción núcleo, y las clases de ayuda para que se pueda trabajar con canvas de manera mucho más fácil.

- **PixiJS [18].** El objetivo de PixiJS es proporcionar una biblioteca 2D rápida y ligera que funcione en todos los dispositivos. Se trata de un renderizador que permite a todos disfrutar del poder de aceleración de hardware sin conocimiento previo de WebGL.

- **GameMaker [19].** GameMaker Studio es un sistema de creación de juegos creado por Mark Overmars en el lenguaje de programación Delphi. Está dirigido tanto a principiantes como a profesionales del desarrollo de juegos por igual, permitiendo la creación de juegos multiplataforma con una reducción tanto en coste como en tiempo de desarrollo.

Capítulo 3. Desarrollo multiplataforma

Una de las cuestiones fundamentales a la hora de desarrollar una aplicación es a qué plataforma dirigirla. Actualmente, la cuota de mercado de los smartphones [20] se la reparten las siguientes plataformas: Android, de Google, con 81% de cuota de mercado a nivel mundial; iOS de Apple, con una cuota de mercado del 12.9%, Windows Phone, de Microsoft, con un 3.6%; BlackBerry OS, de BlackBerry con un 1.7% y dejando a otros sistemas operativos con un 0.6%.

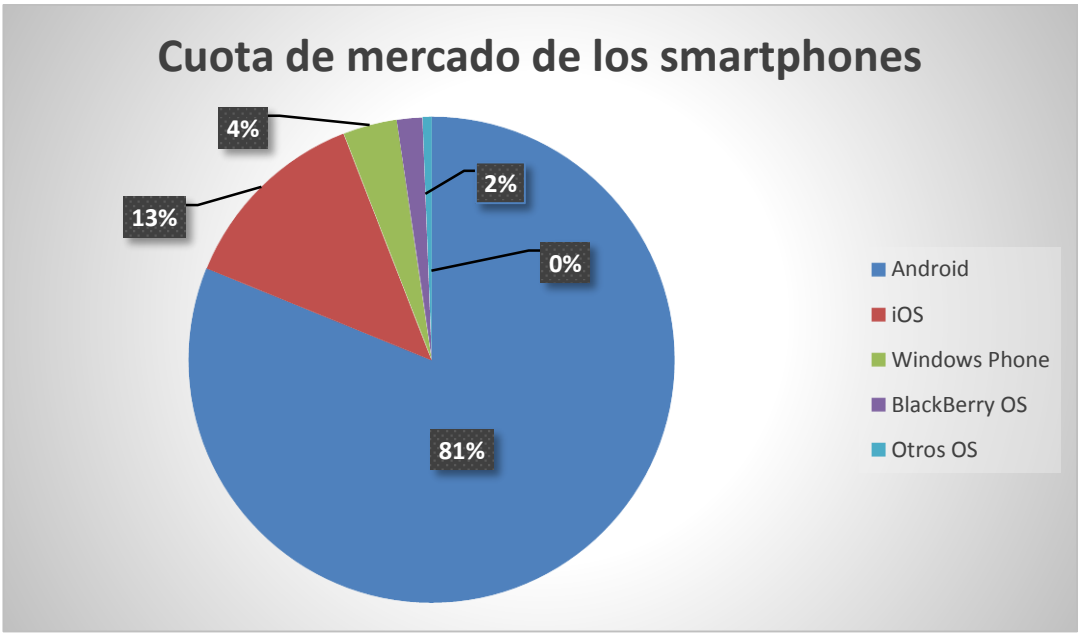

Figura 2: Gráfico de la cuota de mercado

Además, podemos encontrar numerosas herramientas basadas en HTML5, JavaScript y CSS3 para el desarrollo multiplataforma. Destacamos PhoneGap y Appcelerator Titanium por ser las que presentan un uso más extendido entre los desarrolladores. Sin embargo, estas dos no son las únicas que trataremos a continuación:

3.1 PhoneGap

PhoneGap [21] es un framework para el desarrollo de aplicaciones "nativas" de sistemas operativos móviles, haciendo uso de tecnologías web como HTML5, CSS3 y JavaScript que permite desarrollar para Android, iOS, Windows Phone, BlackBerry OS, Web OS, Symbiam y Bada.

Una aplicación PhoneGap consta de una serie de páginas web que están almacenadas y empaquetadas dentro de una aplicación móvil y son visualizadas con un navegador web, con acceso a la mayoría del hardware del móvil (cámara, acelerómetro, notificaciones, etc) mediante una API que proporciona. Además, permite maquetado mediante DOM.

Esto significa que realmente las aplicaciones resultantes son "híbridas" y no "nativas", sin embargo, el rendimiento es bueno, ya que la API se conecta al sistema operativo usando el código nativo del sistema huésped a través de FFI (Foreign Function Interface) en JavaScript.

3.1.1 Adobe PhoneGap Build

Phonegap cuenta con su servicio de construcción en la nube llamado Adobe PhoneGap Build, en el cuál, simplemente deberemos subir nuestros archivos HTML5, CSS y Javascript para que dicho servicio haga el trabajo de compilar el código por ti.

3.2 Appcelerator Titanium

Appcelerator Titanium [22] es un framework que emplea el lenguaje Javascript para desarrollar aplicaciones de escritorio y para móviles (iOS, Android y Blackberry). Incluye un SDK de código abierto que cuenta con unos 5000 dispositivos y APIs de sistemas operativos. Además, proporciona su propio IDE basado en Eclipse, Titanium Studio, con el que poder crear

los proyectos y gestionar todos los recursos del mismo y, en teoría, genera aplicaciones nativas en lugar de aplicaciones que se ejecutan dentro de un navegador, por lo que el rendimiento es bastante bueno.

Al contrario que ocurre con PhoneGap, Titanium no maqueta usando el DOM, sino que se interactúa mediante JavaScript con su propia API.
Esta API provee elementos de interfaz de usuarios basados en la representación de los respectivos controles nativos de las distintas plataformas, es decir, la API hace de intermediario entre la aplicación javascript y los controles nativos del sistema, de ahí que se consiga un mayor rendimiento.

3.2.1 Titanium Cloud Services

Titanium ofrece su propia plataforma para BaaS (Backend as Service), que ofrece una manera fácil y rápida de conectar las aplicaciones móviles. Así permite servicios como enviar notificaciones, actualización de estados, almacenamiento de fotos e integración en redes sociales.

3.3 Corona SDK

El framework Corona SDK [23] permite el desarrollo de aplicaciones y juegos 2D en el lenguaje LUA. Entre sus características se encuentra: acceso a diferentes periféricos del dispositivo, dispone de motor físico interno, permite también utilizar elementos de interfaz nativos de las diferentes plataformas y permite trabajar con elementos gráficos de una manera muy sencilla. Cuando finaliza el proceso de desarrollo se puede llevar a cabo la compilación de la aplicación tanto para android, como iphone/ipad, kindlefire, el lector de libros electrónicos nook.

Realmente presenta múltiples ventajas, como pueden ser la integración automática con OpenGL-ES, el desarrollo multiplataforma, buen rendimiento debido al uso de hardware de aceleración, posee controles nativos para el acceso al dispositivo y finalmente, el empleo de lenguaje LUA, que es potente y fácil de aprender.

Otra de ellas es que podemos utilizarlo totalmente gratis. Basta con llenar un formulario y obtendremos una versión trial que contiene todas las funcionalidades y que podemos usar por tiempo ilimitado para desarrollar nuestros juegos.

Sin embargo, presenta como desventaja que si queremos darle uso comercial es cuándo tendremos que comprar la licencia que cuesta $192 dólares al año en la versión Basic y que nos da acceso para realizar compras dentro de la aplicación desarrollada (conocida comúnmente como “in-app purchase”).

Además como Ansca, la compañía detrás de Corona, no es oficialmente parte ni de Apple ni de Android, hay ciertas funcionalidades que pueden no estar disponibles en la última versión del SDK nativo.

3.4 CocoonJS

CocoonJS [24] es una plataforma que te permite testear, acelerar, desplegar y monetizar tus aplicaciones y juegos HTML5 en todos los dispositivos móviles con diversas funcionalidades.

Dado que el rendimiento es clave en una aplicación web móvil y para el desarrollo de juegos, CocoonJS fue diseñado desde el principio para acelerar todos los aspectos de la ejecución de HTML5. Dependiendo del tipo de proyecto, ofrece entornos diferentes para cada uno de ellos:

- **Canvas+.** Para proyectos de juegos basados en HTML5 y canvas.
- **Webview+.** Para proyectos de aplicaciones de propósito general o juegos basados en DOM.

Su filosofía de trabajo es bastante simple y se puede apreciar en la siguiente imagen, extraída directamente de la página web de CocoonJS:

Figura 3: Cómo usar CocoonJS [25]

Sigue tres sencillos pasos:

1. Desarrollar la aplicación o el juego HTML5 de la forma deseada.
2. Testear cómo funciona directamente en el dispositivo mediante su aplicación CocoonJS Launcher App que funciona tanto para iOS como para Android.
3. Subir el código a la plataforma basada en la nube CocoonJS Cloud System, donde se creará la aplicación final para distribuir en las diferentes tiendas.

3.4.1 CocoonJS Cloud System

CocoonJS ofrece un compilador en la nube fácil de usar en donde se podrán seleccionar y configurar las diferentes plataformas para desplegar una aplicación en ellas.

Además permite configurar servicios extra como monetizar, analizar y ajustar en función de la plataforma.

Capítulo 4. Diseño e implementación.

En este capítulo describiremos la fase de diseño y de implementación de la aplicación móvil.
Veremos las decisiones tomadas para el desarrollo en base al estudio multiplataforma realizado y en función de los requisitos de la aplicación.
También ahondaremos en la aplicación en sí, su estructura, así como el resultado del estudio de las posibles funcionalidades planteadas en el Objetivos y requisitos.

4.1 Elección del entorno de desarrollo

Tras valorar el análisis expuesto en el **Desarrollo multiplataforma para móviles.** En un mundo donde debemos adaptarnos continuamente, realizar un código que podamos reutilizar para las diferentes plataformas es el mejor ejemplo de adaptación. Además de poder llegar a más dispositivos móviles, algo que interesa bastante, el desarrollo multiplataforma (en este caso para Android e iOS aunque es extensible en el futuro) permite ahorrar en tiempo, costo y en mantenimiento en función de los requisitos de la aplicación. de esta memoria, queda justificada la elección de dirigir nuestra aplicación a Android e iOS, dado que son los Sistemas Operativos que más cuota de mercado abarcan.

En cuanto a las plataformas presentadas, he de justificar primero la inclusión de CocoonJS en ellas, a pesar de que no es un framework propiamente dicho.

El motivo principal por el cual se incluyó su análisis es que puesto que se va a desarrollar el prototipo de un juego, el rendimiento toma un aspecto vital,

por lo que la relevancia de CocoonJS y su entorno acelerado cobra sentido. Esto lo consigue porque emplea una solución híbrida que combina HTML5, CSS y JavaScript y bibliotecas de OpenGL para aprovechar el hardware nativo.

También cumple uno de los objetivos de este proyecto, el destinado al estudio de la transformación posterior del juego en aplicación, cosa que podremos hacer con su plataforma en la nube.

CocoonJS, además de ser un nuevo aporte a la lista de herramientas para la elaboración de juegos multiplataforma, no solo cumple con este objetivo sino que es agnóstico a la plataforma que se elija. No te condiciona tanto como ocurre con el resto, ya que te permite la libertad de desarrollar empleando el framework o game engine que se desee.

Esto abre un nuevo camino, dado que mientras se cumpla el paradigma de HTML + CSS + JavaScript, puedes testear y desplegar el juego, lo que hace incluso que ni siquiera tengas la necesidad de emplear alguna de ellas, Por este mismo motivo, y dado que el prototipo es un juego de puzzles de una no muy elevada complejidad, he decidido que esta será la vía de desarrollo.

Además, CocoonJS te da diferentes APIs, tanto para acceder al hardware del dispositivo, como soporte para WebGL, sonidos, notificaciones, etc.

Por tanto, será un juego basado en Javascript, que emplee las librerías o frameworks que se estimen oportunos para el desarrollo. Esta vía sólo sólo podrá volver a ser replanteada en el caso en que se vea que CocoonJS no baste para cubrir todas las necesidades y/o requisitos de la aplicación.

Si se diese el caso, se emplearía Phonegap, por las facilidades que brindan sus APIs frente a Appcelerator Titanium, a pesar de que está pensada para otro tipo de aplicaciones más que para juegos [26].

Corona SDK también queda descartada pero por motivos diferentes, dado que emplea el lenguaje de programación LUA, quedándose fuera de los objetivos del proyecto al no emplear tecnologías web.

4.2 Desarrollo de Puzzleyap

El código desarrollado del juego Puzzleyap se encuentra en el siguiente repositorio público de github: https://github.com/DavidHerbetULL/TFG-PuzzleYap

En la carpeta "CocoonJSLauncher" se encuentran las diferentes iteraciones del prototipo del proyecto.

4.2.1 Desarrollo mediante CocoonJS

Lo primero a nombrar en el desarrollo es que, puesto que hemos optado por emplear la filosofía de CocoonJS, todo el código desarrollado deberá ser testeado en la aplicación CocoonJS Launcher, lugar donde se verificará que todo funcione correctamente.

Cómo se vea nuestro juego en dicho Launcher determinará como se visualizará finalmente la aplicación, por lo que esta será nuestra herramienta fundamental. Si queremos probar que una determinada librería sea compatible con nuestro proyecto habrá que pasar el visto bueno del launcher, si no, no podremos utilizarla.

Para esto simplemente deberemos de instalarnos el launcher de CocoonJS en nuestro dispositivo móvil, crear un fichero comprimido en zip con el código de nuestra aplicación y guardarlo en la tarjeta SD interna del dispositivo. Una vez seguidos estos pasos, ya podremos visualizarlo desde el launcher.

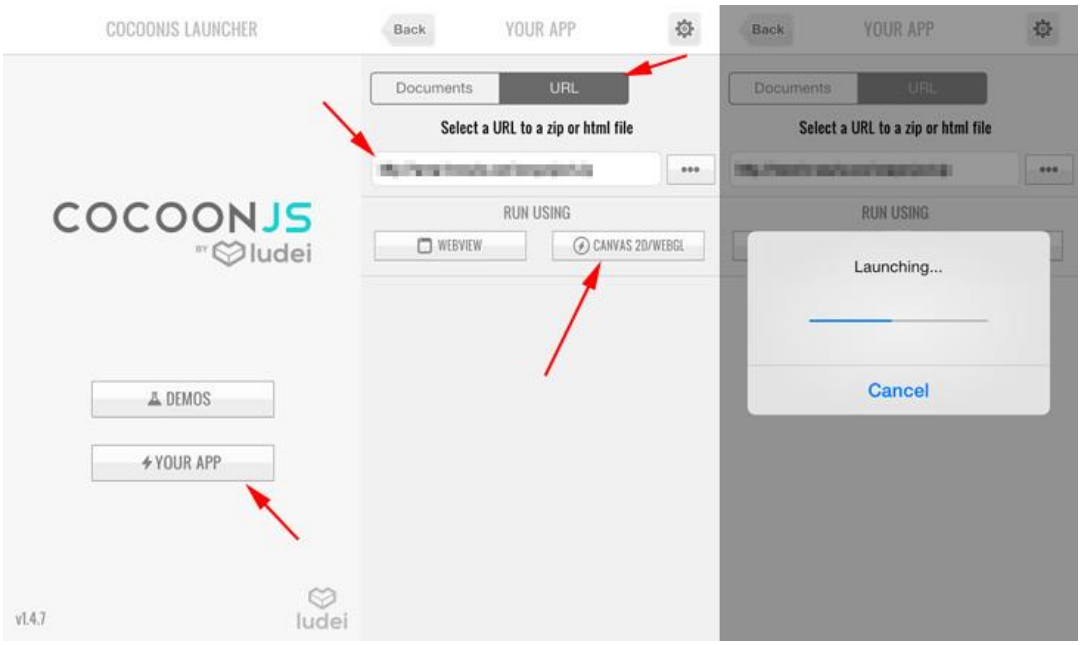

Figura 4: CocoonJS Launcher

Dado que nuestro prototipo seguirá un curso iterativo, cada versión del prototipo tendrá asociado un fichero zip que se generará para testear su funcionalidad en el launcher.

Los dispositivos empleados para testear la aplicación han sido tres:

1. Samsung Galaxy Mini 2 (Android 2.3.6)
2. iPad 2 (iOS 7.1.2)
3. Samsung Galaxy Tab 4 (Android 4.4.2)

4.2.2 Desarrollo de juegos 2D

En esta fase de inicio, donde se debe sentar la base y crear una buena estructura, debemos plantearnos qué es realmente un juego para conocer las peculiaridades de su desarrollo.

Un juego, al fin y al cabo, no es más que un bucle infinito controlado, el cual, se encuentra continuamente comprobando valores para actualizar y pintar en la pantalla en función de ellos.

Cada pantalla del juego representará un estado, a partir del cual, se podrán realizar acciones o transitar a otros estados para cumplir el objetivo del juego.
Para este cometido se creará el motor de estados gestionado a través de la pila del juego, conocido en inglés como State Stack Engine.

El bucle principal del juego se llamará cada pocos milisegundos y delegará sus funciones de actualizar y pintar a los estados que se introduzcan en la pila. Cada estado a su vez tendrá los suyos propios, los cuales delegarán en los métodos de cada componente para actualizarse y pintarse, de este modo se logra una mayor gestión y control de los objetos, pudiendo tener una clase que se encargue en la interfaz de usuario de los botones de la aplicación

Cada objeto de estado llevará asociado una función de **entrada**, la cual se llamará una vez al principio para realizar todas las tareas de preconfiguración del estado; una función de **actualizado** y una de **renderizado**, las cuales serán llamadas constantemente en ese orden para actualizar los valores correspondientes y pintarlos en la pantalla mientras el estado esté activo y, finalmente, una función de **salida**, encargada de limpiar el estado.

Opcionalmente se pueden agregar dos funciones más, una que **pause** el juego y otra que lo **reanude**. No obstante para nuestra aplicación estos dos últimos no se van a usar realmente porque no son necesarios.

La siguiente figura refleja el diagrama de estados de Puzzleyap:

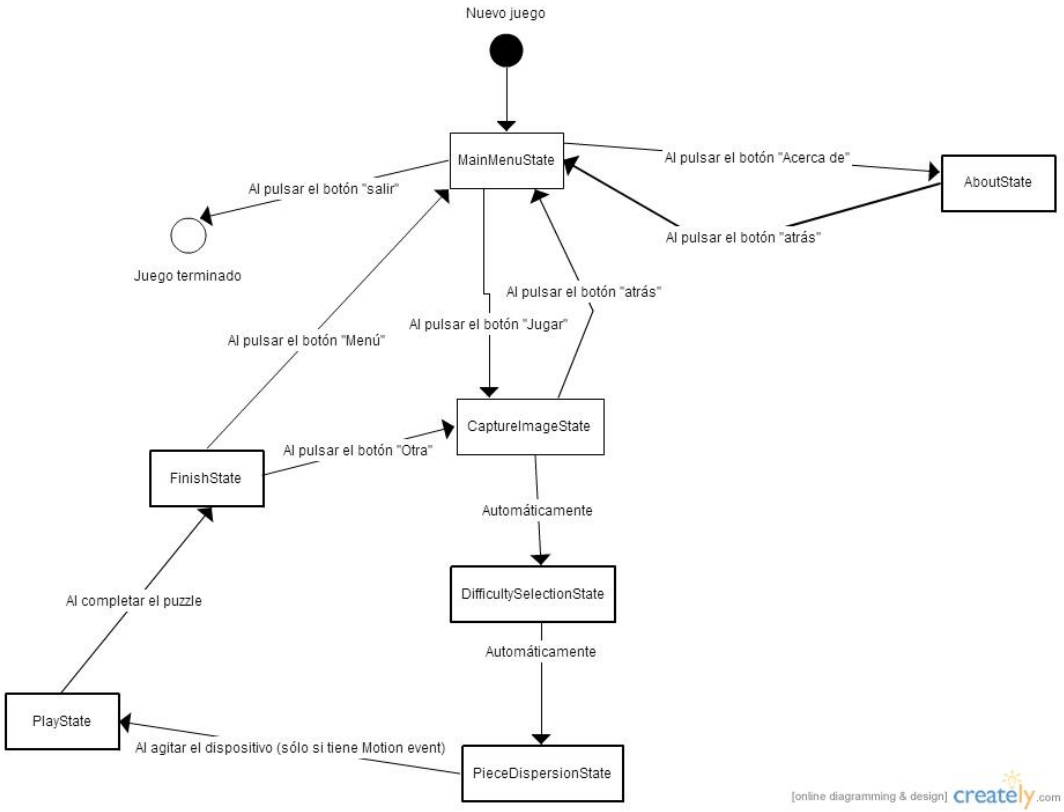

Figura 5: Diagrama de estados de Puzzleyap

Las clases que se han desarrollado se muestran en las figuras a continuación:

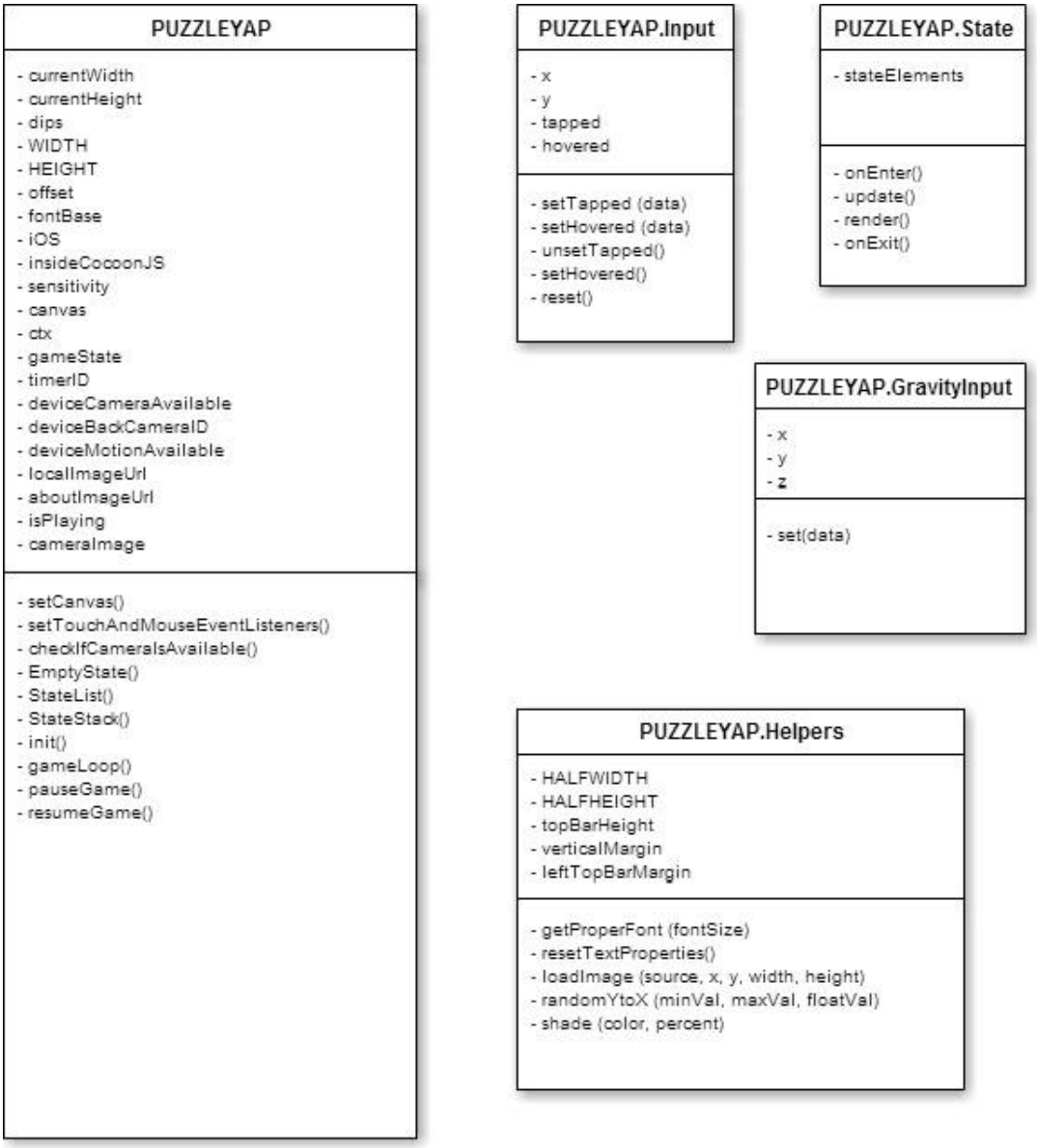

Figura 6: Clases de Puzzleyap 1

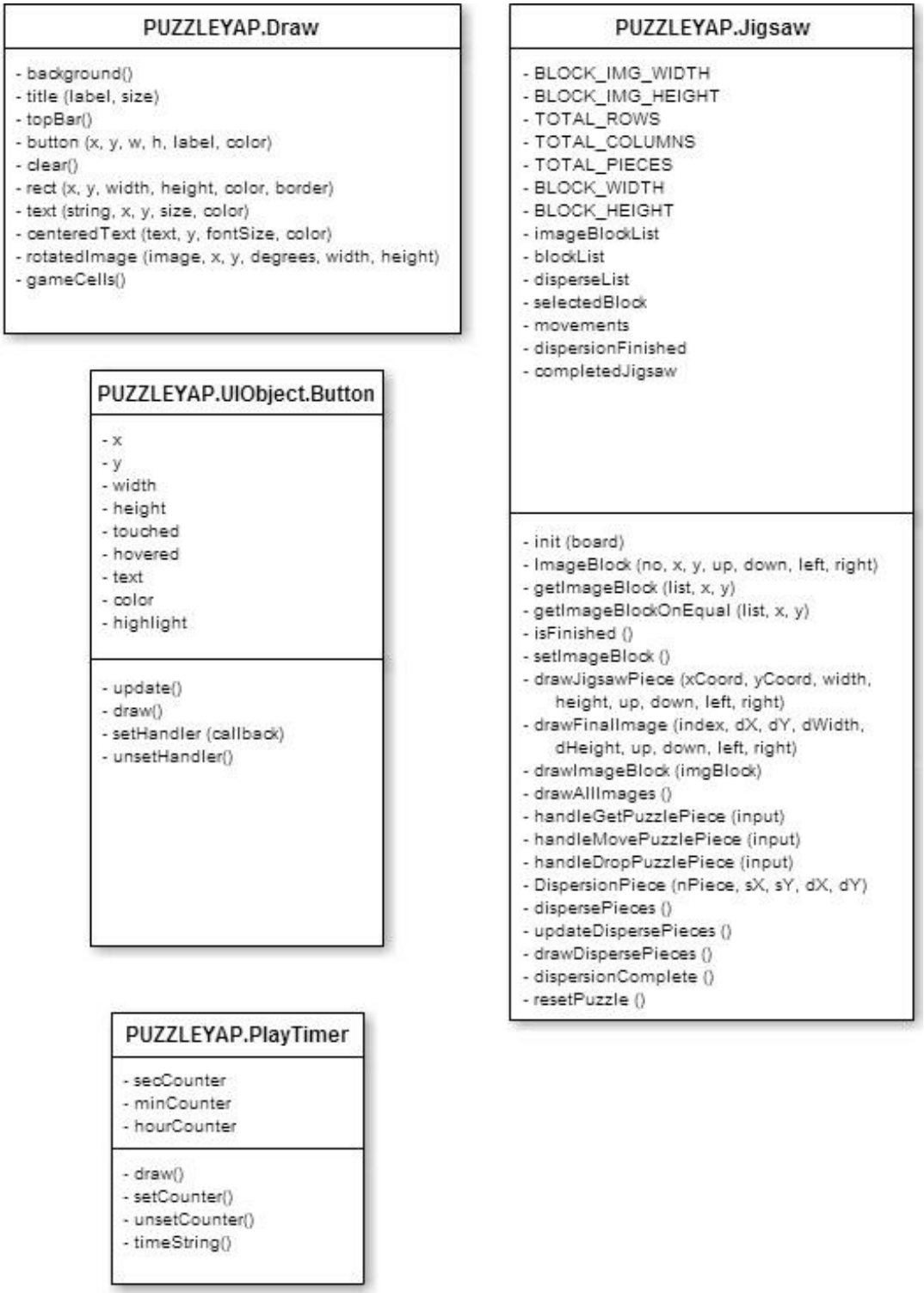

Figura 7: Clases de Puzzleyap 2

4.2.3 Captura y tratamiento de imágenes

Para este propósito contamos con la extensión de Cámara de CocoonJS, que es capaz de acceder a la cámara del dispositivo en cuestión.

Parecía prometedora pero nos encontramos con que no permite tomar fotografías desde la cámara en sí. Nos da un objeto de tipo imagen que se va actualizando constantemente en función de la cámara.

Este supuso el primer impedimento real de cara al proyecto, dado que tener una API que nos dé acceso a la cámara y no poder capturar una imagen no era lo esperado, además de ser uno de los objetivos que nos habíamos propuesto.

No obstante, dado que la imagen que se va actualizando se muestra en el canvas, podemos solventar el problema mediante el método toDataURL() de la API de canvas [29]. Esta función realiza una captura de la pantalla y devuelve una URL que contiene la representación de la captura tomada en un formato especificado (por defecto PNG), con lo que podemos emular sin problema el efecto de sacar una foto.

En un primer momento se planteó la posibilidad de enviar la imagen a un servidor en http://banot.etsii.ull.es/, procesarla y enviarla devuelta a la aplicación para la generación del puzzle por si las operaciones eran muy pesadas. Sin embargo, como veremos más adelante, no fue necesario gracias al tratamiento de las imágenes en canvas.

4.2.4 Generación del puzzle

Para la generación del puzzle, se consideró que la mejor opción era emplear alguna librería o framework que dotase de facilidades para no tener que

gestionar la generación de las piezas y su interfaz de arrastre (drag and drop en inglés).

Asumiendo esto se intentó usar el framework Paper.js [30]. Se trata de un framework open source para la manipulación de vectores gráficos, lo cual nos daría facilidades a la hora de crear las piezas con forma jigsaw para el puzzle. Sin embargo, el Launcher de CocoonJS nos impidió su uso debido a incompatibilidades.

También se intentó incorporar el framework Kinetics.js [31], el cual está basado en JavaScript y canvas de HTML5 que permite animaciones de alto rendimiento, transiciones, capas, filtros, etc.
Este framework, más potente y completo que Paper.js, permite dibujar elementos en el canvas, moverlos, escalarlos y rotarlos independientemente de otras capas para mejorar el rendimiento.
Lamentablemente, el Launcher de CocoonJS nos volvió a impedir su uso.
Llegados a este punto uno se cuestiona bastante la característica de CocoonJS de "desarrolla de la forma en que desees". Esto significará que habrá que realizar la generación del puzzle sin ningún tipo de soporte, mediante JavaScript, de lo contrario habríamos llegado a un nuevo punto crítico en el que no podríamos continuar.
El primer paso para generar el puzzle es el fragmentado. Gracias al método drawImage() de canvas, esta tarea ha resultado ser bastante trivial, dado que el método permite dibujar la imagen y fragmentos de la misma. Esto se hace situándonos en el eje de coordenadas (x, y) de la imagen e indicando el tamaño del bloque del puzle. La siguiente figura muestra cómo mediante una única imagen podemos sacar las diferentes piezas con drawImage:

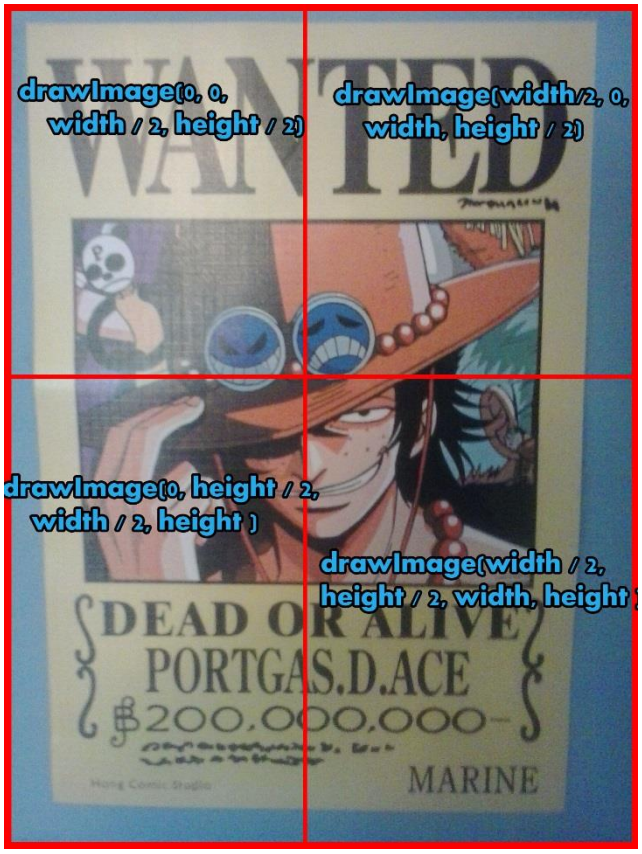

Figura 8: Ejemplo de uso de drawImage()

La complejidad reside en dar la forma de piezas al puzzle. Para ello deberemos de emplear el concepto de recorte o clipping. La idea es que mediante una ruta o path, definimos un camino en canvas que especifica que las futuras operaciones sólo deben afectar a la zona de dicha ruta.

Ya que el canvas es transparente, si creamos una ruta con la forma de una pieza jigsaw y posteriormente dibujamos la imagen, conseguiremos que de la imagen sólo se muestre la parte que se encuentre dentro de nuestra ruta de pieza jigsaw, logrando nuestro objetivo. La siguiente figura ilustra el efecto:

Figura 9: Clipping

En la figura la zona del color rojo corresponde al path que hemos creado con la forma del puzzle, mientras que la parte de los bordes amarillos muestra el tamaño de la imagen real, sólo que para nosotros únicamente es visible la zona del roja del path.

Con esta idea en la mente, se fragmenta la imagen en diferentes piezas jigsaw aleatorias, mediante un algoritmo que deberá tener en cuenta lo siguiente:

- Cada pieza tiene cuatro lados.
- Cada uno de esos lados puede tener 3 diferentes estados:
    - **Montaña.** Representa la pieza con hueco cóncavo, representado por el valor 1.
    - **Valle.** Representa la pieza con huevo convexo, representado por el valor -1.
    - **Plano.** Representa la pieza sin hueco, representado por el valor 0.
- Los bordes del puzzle tienen estado plano.

- Empezando por la primera pieza, se decidirá aleatoriamente si el lado derecho y el de abajo tienen montaña o valle y a su pieza contigua le asignaremos el contrario de la selección tomada.

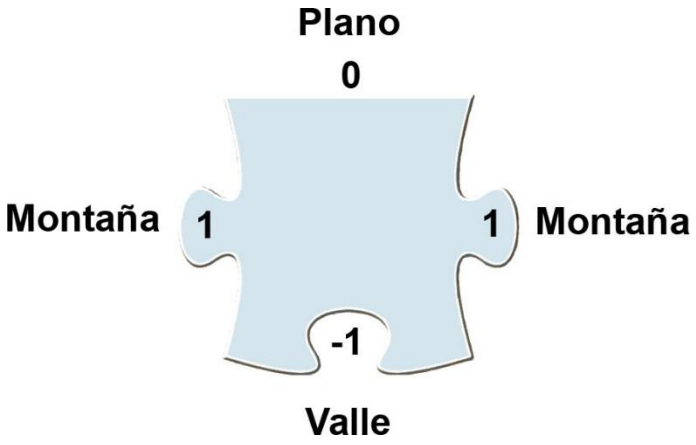

Figura 10: Pieza jigsaw

Una vez hecho esto ya tenemos todos los elementos necesarios para construir el puzzle. Se crea un array de piezas de puzzle y otro de posiciones a las cuales les asignaremos un número de pieza para saber la correspondencia y posición final. Cuando todas las piezas del puzzle se encuentren en las posiciones correspondientes del array de posiciones, se dará el puzzle por concluido.

Se ha empleado la librería Lo-Dash [27], la cual es una biblioteca de utilidades de bajo nivel que ofrece consistencia, personalización y rendimiento, además de múltiples funcionalidades de ayuda en Javascript. Principalmente he empleado los siguientes métodos:

- **__.each(collection, [callback=identity], [thisArg]).** Itera sobre los elementos de una colección, ejecutando un callback por cada elemento.
- **__.isUndefined(value).** Comprueba si el valor es "undefined".
- **__.extend(object, [source], [callback], [thisArg]).** Asigna las propiedades del objeto fuente al objeto de destino, las fuentes

posteriores sobrescribirán las asignaciones de propiedad de las fuentes anteriores.

- **__.first(array, [callback], [thisArg]).** Obtiene el primer elemento o los primeros n elementos de un array.

También se intentó usar la librería Zebkit [28], que permite el renderizado de componentes de interfaz de usuario sobre canvas. Sin embargo resultó ser incompatible ya que el Launcher de CocoonJS lo rechazó, por lo que los elementos de la interfaz de usuario han sido creados mediante canvas directamente.

Ahora sólo nos queda implementar su resolución, para ello emplearemos los escuchadores (listeners en inglés) de canvas para los Touch Events (o Mouse events en caso de no estar en dispositivo móvil):

- **onTouchStart/onMouseDown:** Este evento se lanza cuando el usuario toca o hace un click en la pantalla. En este listener comprobaremos cuando el usuario pulse sobre la pantalla que lo haya hecho sobre una de las piezas del puzzle, para colocarla en estado seleccionada.
- **onTouchMove/onMouseMove:** Este evento se lanza cuando el usuario que ha iniciado un click/toque arrastra o desliza el cursor por la pantalla. En este listener actualizaremos la posición de la pieza (coordenadas x, y) conforme se deslice la pieza por la pantalla.
- **onTouchEnd/onMouseUp:** Este evento se lanza cuando el usuario deja de tocar o termina el click de la pantalla. En este listener comprobamos que la posición en la que vamos a soltar la pieza sea una posición válida, es decir, que otra pieza del puzzle no la haya ocupado, para colocarla en la casilla.

4.2.5 Acelerómetro

Como bien se definió en los objetivos del 1.2.Uno de los aspectos a investigar era la posibilidad de que el puzzle comenzase a resolverse tras agitar el dispositivo y para ello se necesita acceso al hardware del dispositivo, concretamente, al acelerómetro.

Los eventos encargados de la aceleración son los llamados Device Motion [32], gestionan los datos de la aceleración en un eje de coordenadas (x, y, z) y son medidos en m/s^2.

Debido a que algunos dispositivos no poseen el hardware para excluir el efecto de la gravedad, el evento maneja dos propiedades: aceleración incluyendo la gravedad y aceleración sin incluirla.
Teniendo esto en cuenta, sólo nos hace falta comprobar si el dispositivo soporta eventos Device Motion mediante la propiedad "window.DeviceMotionEvent". Una vez que tengamos soporte, podremos agregar los listeners correspondientes, que se encargarán de estar continuamente actualizando los datos de aceleración conforme movamos el dispositivo y definiremos un umbral sobre el cual, si la suma de los valores absolutos de cada componente (x, y, z) lo sobrepasa, se considerará que se ha "agitado" el dispositivo.

Esto que se ha descrito es bastante sencillo, sin embargo en la práctica no lo ha resultado ser tanto, dado que se ha detectado que un dispositivo pueda decir que si soporta Device Motion, pero a la hora de la verdad no lo implemente, ni maneje los eventos, como sucede con el dispositivo Samsung Galaxy Tab 4.
No obstante, el caso se puede identificar, ya que al actualizar las coordenadas (x, y, z) mediante el listener, si no se ha producido ningún cambio en las componentes (y eso es algo imposible, dado que aunque dejes

el dispositivo en reposo sobre un superficie, el valor de las coordenadas siempre va a oscilar aunque sea un poquito no llegando a ser nunca cero), se puede asumir que el dispositivo dice que soporta Motion Events, pero realmente no lo hace.

Otro problema añadido es que los diferentes dispositivos pueden tomar valores muy diferentes de las coordenadas (x, y, z), como es en el caso de los otros dos dispositivos restantes, el Samsung Galaxy Mini 2 y el iPad 2.
Debido a esto se han tenido que establecer esos umbrales en función de que el dispositivo sea iOS o Android.

Para esta parte se ha creado un algoritmo muy sencillo que dispersa las piezas del puzzle, llevándolas a su posición final aleatoria.
Este estado espera a que se agite el dispositivo para llevarlo a cabo o, si el dispositivo no posee Motion Events, se salta directamente el estado de espera.

4.2.6 Efectos especiales

En la red se encuentran numerosas librerías de JavaScript en lo que se refiere a manipulación y tratamiento de imágenes. Sin embargo, nuestro requerimiento era provocar un efecto en la imagen que fuese de distorsión, más que de cambiar contrastes, tonos de color u alguna cosa de la misma índole.

Se investigaron diversas librerías JavaScript pero solo encontramos una que realmente cumpliese con nuestro propósito, esa librería es glfx.js [33].Se trata de una librería de efectos de imagen impulsada por WebGL, que emplea la propia tarjeta gráfica para crear efectos que serían imposibles de aplicar en tiempo real empleando únicamente JavaScript.

CocoonJS soporta WebGL, lo que hace que se rompa la primera barrera que podríamos encontrarnos, sin embargo, no se ha podido usar la librería glfx.js.
El hecho de incluir la librería en nuestro proyecto sin realizar ningún cambio (es decir, incluir el script con el código de la librería) hace que la pantalla se quede en negro y no haya canvas que visualizar. Seguramente esto signifique que en alguna parte del código se sobreescriben elementos o se crean incompatibilidades entre sí, a pesar de que CocoonJS soporte WebGL.

Esto se hace evidente, dado que testeando la aplicación desde el navegador web en el ordenador, no presenta problema ninguno, el Launcher de CocoonJS es el que pone de manifiesto la incompatibilidad.

Otra opción por la que se optó, dado que CocoonJS es compatible con WebGL, utilizar alguna librería puramente basada en WebGL para los efectos que deseamos. Sin embargo, no se encontró ninguna que cumpla con esos requisitos.

Finalmente se ha recurrido a escoger uno efectos de la librería glfx.js, que se adecuase a lo que teníamos pensado, en este caso, el de torbellino o swirl en inglés, e implementar la rotación de píxeles manual [34] en JavaScript.

El efecto swirl básicamente consiste en girar los píxeles de alrededor del centro una imagen hacia la izquierda o hacia la derecha creando con ello una especie de vórtice. El efecto consiste en la rotación individual de los píxeles de la imagen en un ángulo que es proporcional a su distancia desde el centro.

Lamentablemente, tras haber implementado el efecto, nos encontramos nuevamente con la última barrera: realizar esa operación de rotación de píxeles sobre la imagen en el canvas es demasiado costosa y colapsa el dispositivo, sea tablet o móvil, por lo que se ha dejado el efecto sólo para la versión web de la misma. La siguiente figura muestra cómo se vería en la pantalla final desde el navegador web:

Figura 11: Efecto de torbellino o swirl

4.2.7 Pantallas de Puzzleyap

En este apartado se mostrarán de la aplicación final, las cuales se corresponden con el diagrama de estados expuesto en la 4.2.2.

Figura 12: Inicio

Figura 13: Acerca de

Figura 14: Capturar imagen

Figura 15: Seleccionar dificultad

Figura 16: Agitar el dispositivo

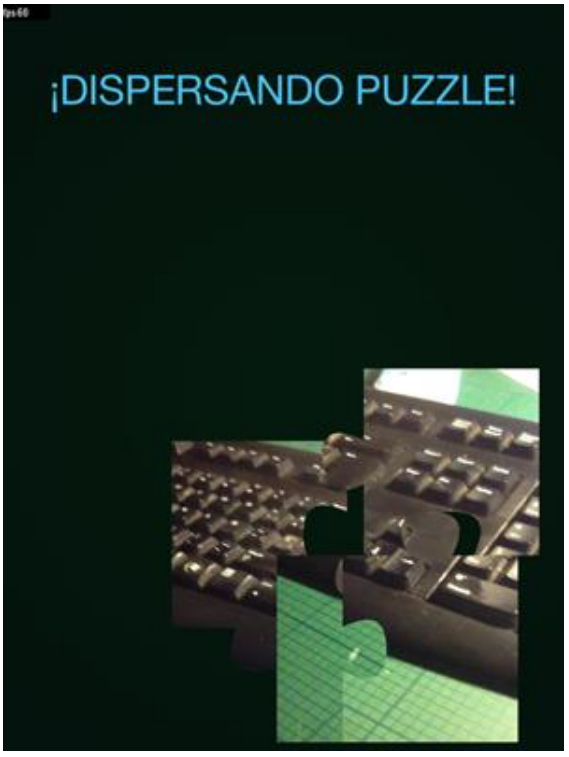

Figura 17: Dispersando el puzzle

Figura 18: Resolviendo el puzzle

Figura 19: Pantalla final

4.2.8 Resumen de librerías

Las siguientes tablas se reflejan en resumen todas las librerías y/o framework que se han usado o se han intentado usar durante el transcurso de este proyecto:

Librería	Descripción	¿Utilizada?	Problema
Lo-Dash	Utilidades y funcionalidades para JavaScript	Si	--
glfx.js	Efectos de imagen mediante JavaScript impulsada por WebGL	No	Incorporar la librería hace que toda la aplicación se cuelgue en el CocoonJS Launcher

Tabla 2: Librerías

Framework	Descripción	¿Utilizado?	Problema
PaperJS	Ofrece un escenario gráfico limpio y funcionalidad para crear y trabajar con vectores gráficos y curvas de Bézier	No	Incompatibilidad con el CocoonJS Launcher
KineticJS	Permite dibujar elementos en canvas, moverlos, rotarlos y escalarlos. También permite transiciones, anidaciones en nodo, almacenamiento en caché y manejo de eventos.	No	Incompatibilidad con el CocoonJS Launcher

Tabla 3: Frameworks

4.3 Transformación en aplicación

Finalmente una vez desarrollado el juego y testeado en el Launcher, se debe de subir el archivo zip resultante al sistema de compilación en la nube de CocoonJS.

Para poder compilar con las extensiones de CocoonJS, es necesario ser usuario Premium. Dado que el proyecto se fundamenta en una de esas extensiones, la de la cámara, para poder sacar la fotografía, ha sido necesario solicitar una cuenta Premium, cosa que nos han concedido.

Para añadir un proyecto en el sistema de compilación en la nube, se deberán proporcionar una serie de recursos, iconos e imágenes, que serán requeridos en función de la plataforma dirigida y configurar el nombre del proyecto (Puzzleyap), su Bundle Id (com.dherbet.puzzleyap), así como la orientación que empleará en el dispositivo. Por si no cuentas con dichos recursos, CocoonJS cuenta con un set predefinido por defecto que puedes usar libremente.
Tras proporcionar lo anterior y añadir el archivo zip con el código del proyecto, éste se envía a una cola de compilación a la espera de ser completado.

A partir de aquí, será necesario realizar una serie de pasos en función de si el destino es Android o iOS.

4.3.1 Creación de un APK instalable para Android

La plataforma de CocoonJS te devuelve un archivo archivo zip tras compilar para Android que contiene dos ficheros APK: uno firmado con una "debug keystore" y otro sin firmar.

Generalmente lo que se obtiene de la compilación de un proyecto es un APK sin firmar, el cuál deberemos firmar con una "debug keystore" si lo que queremos es instalar la aplicación en un dispositivo, o con una "release keystore", si lo que deseamos es subir la aplicación a Google Play.

Dado que nuestro objetivo es instalar el juego en nuestro dispositivo, y que nos han dado un APK ya directamente firmada con la release keystore que nos haría falta, simplemente tenemos que copiar el archivo firmado a la SD, habilitar la opción de instalar fuentes desconocidas (En Ajustes → Aplicaciones) y seleccionar el archivo desde el dispositivo para comenzar con la instalación.

Con esto ya podremos comenzar a utilizar nuestra aplicación en Android, y sin tener ninguna necesidad de instalar su SDK.

Si quisiéramos desplegar la aplicación en Google Play, entonces sí que tendríamos que instalar el SDK de Android, con Java 6 (con Java 7 da problemas en CocoonJS) y utilizar las utilidades Keytool y Jarsigner que incorpora el SDK para firmar. Se puede realizar mediante Android Studio, el plugin ADT para Eclipse o manualmente desde la línea de comandos, como vamos a ver a continuación:

1. Generamos la realease keystore con Keytool:
   $ keytool -genkey -v -keystore my-release-key.keystore
   -alias alias_name -keyalg RSA -keysize 2048 -validity 10000
2. Firmamos con Jarsigner el archivo proporcionado sin firmar:
   $ jarsigner -verbose -keystore
   PATH/TO/YOUR_RELEASE_KEY.keystore -storepass
   YOUR_STORE_PASS -keypass YOUR_KEY_PASS
   PATH/TO/YOUR_UNSIGNED_PROJECT.apk
   YOUR_ALIAS_NAME
3. Alineamos el archivo:
   $ zipalign -v 4 PATH/TO/YOUR_SIGNED_PROJECT.apk
   PATH/TO/YOUR_SIGNED_AND_ALIGNED_PROJECT.apk

Tras haber seguido estos pasos tenemos un APK firmado listo para subir a Google Play.

4.3.2 Usar XCode Project generado por el compilador en la nube para publicar el proyecto en iOS

Para poder usar el XCodeproject, hay una serie de pre-requisitos que deberán cumplir:

- Se necesita un ordenador Mac.
- Se necesita la última versión instalada de XCode (junto con la dependencia Command Line Tool).
- Se necesita una cuenta de desarrollador de Apple.

Para poder obtener una cuenta de desarrollador de Apple, se ha de adquirir una licencia, la cual es de pago. La única excepción es pertenecer a una universidad que esté suscrita a su programa de desarrolladores.

Gracias a que la Universidad de la Laguna está adscrita a ese programa, puedo compilar mi proyecto para iOS, sin eso sería imposible poder compilar.

Para el proceso mi directora me dió de alta en el programa de desarrolladores (http://developer.apple.com) y me cedió un Mac y un iPad temporalmente.

Como ese programa tiene requisitos para los miembros, ella misma ha tenido que los pasos que se describirán a continuación.

El primer paso será crear un ID de Apple para la aplicación, para ello se deberá suministrar un nombre, una Bundle Seed ID de las que te

proporcionan y un Bundle Identifier. Este último es muy importante y debe de ser el mismo Bundle ID de la plataforma en la nube de CocoonJS, es decir, com.dherbet.puzzleyap.

El segundo será crear un provisioning profile. Aquí podemos elegir entre dos tipos de perfiles, en función de si queremos subir nuestra aplicación a la App Store o instalarla en nuestros dispositivos, siendo los perfiles de AdHoc distribution y AppStore distribution respectivamente.
A diferencia que en el paso con Android, si queremos testear dispositivos con el perfil AdHoc distribution, deberemos añadirlos al portal por su UDID, identificador de dispositivo único.

- Para crear un Adhoc distribution profile hace falta suministrar un nombre, un App ID (el de nuestra aplicación creada anteriormente) y un dispositivo o varios sobre los que se quiera testear (el iPad en nuestro caso).
- Para crear un AppStore distribution profile sólo es necesario seleccionarlo.

Una vez llegados a este punto, habrá que ir al portal de desarrolladores de Apple para solicitar un certificado de desarrollo. Para ello necesitas crear un fichero CSR a través de "Acceso a llaveros" en el Mac. Dentro del programa vamos al Asistente para certificados → Solicitar un certificado de una autoridad de certificación. Ahora rellenamos los campos de email, nombre común y seleccionamos la opción de guardar en el equipo.
Con este archivo creado podemos ir al portal y generar nuestro certificado.

A partir de aquí tenemos todo lo necesario para compilar el proyecto. Abrimos Xcode, seleccionamos el archivo de proyecto que nos suministra la plataforme en la nube de CocoonJS, comprobamos que en la pestaña

Product esté marcada la opción iOS Device y ejecutamos Product → Archieve.

Esto firmará la aplicación creando un fichero IPA mediante el cual instalaremos nuestra aplicación (en función del perfil elegido) y el mismo que subiríamos a la AppStore si estuviésemos en esa tesitura.

Como no es nuestro caso, arrastramos el fichero IPA obtenido hacia la librería de iTunes, conectamos el iPad (el dispositivo) y nos aparecerá nuestra aplicación lista para instalar y sincronizar.

Capítulo 5. Conclusiones y Trabajos Futuros

5.1 Conclusiones

En las fases iniciales de este proyecto se definieron una serie de objetivos y requisitos que se debían completar para darlo por superado. Ahora es el momento de repasar dichos objetivos y verificar que se hayan cumplido.

El primero de ellos era investigar las posibilidades de las tecnologías web HTML5 y JavaScript para el desarrollo de juegos 2D. Este objetivo se ha completado satisfactoriamente ya que hemos visto que el uso de HTML5 y JavaScript no sólo es viable, sino que su uso está extendido tanto para navegadores de escritorio como navegadores móviles. Además, comprende un gran abanico de posibilidades a la hora de desarrollar juegos, con muchas opciones a elegir, numerosos motores y librerías con una gran comunidad detrás de la cual abunda mucho soporte e información.

El segundo objetivo era investigar las diferentes plataformas que permiten el desarrollo del juego y su posterior conversión en aplicación para Android o iOS. Este objetivo se ha realizado con éxito y nos ha ayudado a elegir la plataforma de desarrollo que ha sido más conveniente para realizarlo. A excepción de algunos pequeños contratiempos, emplear CocoonJS ha sido completamente satisfactorio y no ha supuesto realmente ningún impedimento a la hora de desarrollar nuestro prototipo. Nos ha permitido llegar al final sin tener la necesidad de replantear el uso de alguna de las otras plataformas para la realización del juego.
No obstante, considero que no se puede llevar al extremo su premisa de "desarrolla de la manera en que se quieras". A esta conclusión se llega tras intentar emplear librerías y frameworks y descubrir que no son compatibles,

llegando a tener que realizar bastantes cosas directamente con JavaScript, en lugar de tener facilidades.

A pesar de ello, conviene mencionar que CocoonJS es compatible con múltiples engines como ImpactJS, BackboneJS, AngularJS, PlayCanvas, y Phonegap entre otros, lo cual, podría indicar que conforme vaya creciendo la plataforma, la compatibilidad con diversos frameworks y librerías irá aumentando. Hasta que ese momento llegue, no se podrá afirmar con seguridad que se puede desarrollar de la manera deseada.

En cuanto al último de los objetivos, consistía en realizar un prototipo de juego de puzzles 2D para Android e iOS empleando las plataformas y las tecnologías investigadas previamente.
Este objetivo también se ha conseguido con éxito, logrando cumplir con el alcance y los prototipos planteados.
Sin embargo, a pesar de haber conseguido realizar los requisitos mínimos para la aplicación, se han encontrado dificultades en la investigación e implementación de los requisitos opcionales, lo cual puede llevar a pensar que alguno pudo haber sido demasiado ambicioso o que se haya fallado parcialmente en su cumplimiento. El poder implementar un modo multijugador se escapa sencillamente a los requisitos de tiempo para la realización de este proyecto, no obstante, se podrá encontrar en el apartado que viene a continuación de 5.2, el fruto de la investigación acerca de cómo llevar a cabo su implementación.

Con respecto al tema de transformar el código realizado en aplicación, el mayor impedimento ha sido a la hora de transformar el proyecto compilado de CocoonJS a iOS, por la parte que le corresponde a Apple, ya que de no ser porque la directora del proyecto disponía de un Mac y porque la Universidad de La Laguna estaba adscrita al plan de desarrolladores de

iOS, no se habría podido haber generado la aplicación como tal para la plataforma de Apple.

En cuanto a valoración personal, decir que he tenido que evolucionar y aprender bastante para realizar este proyecto, del cual en un principio no pensé que pudiese llegar a tal complejidad.
Un ejemplo de esto fue, para mi sorpresa, que el primer prototipo del proyecto, encargado de renderizar una imagen ocupó alrededor de unas 500 líneas de código.
Aun teniendo en cuenta que en él se debe de construir la base fundamental de la aplicación, nunca consideré la posibilidad de que pudiese ocupar tanto.
Lo más probable es que haya sido fruto de mi inexperiencia, pero eso me ha hecho esforzarme más, reforzar los conocimientos que ya poseía y aprender otros nuevos acerca del desarrollo de juegos móviles y sobre todo de las tecnologías utilizadas en el proyecto: CocoonJS y JavaScript.

CONCLUSIONS

In the initial stages of this project a series of objectives and requirements that had to be completed were defined to give this project overcome. Now is the time to review and check if these objectives have been fulfilled.

The first was to investigate the potential of web technologies HTML5 and JavaScript to develop 2D games. This goal has been successfully completed and we have seen that the use of JavaScript and HTML5 is not only viable, but its use is extended for both desktop browsers and mobile browsers. It also includes a large range of possibilities when developing games, with many options to choose from, many engines and libraries with a great community behind which covers much support and information.

The second objective was to investigate the different platforms for game development and subsequent conversion application for Android or iOS. This objective has been successful too and has helped us to choose the most suitable development platform for this project.
However, I think that you can't bear to end his premise "develops the way you want it." This conclusion is reached after attempting to use libraries and frameworks and discover that they are not compatible, reaching many things directly with JavaScript, instead of having facilities.
Despite that, it is worth commenting that CocoonJS supports multiple engines as ImpactJS, BackboneJS, AngularJS, PlayCanvas and Phonegap among others, which could indicate that as the platform grows, support for various frameworks and libraries will increase. Until that time comes, you can't say for sure that can develop in the desired manner.

Last goal consisted to make a prototype of a 2D puzzle game for Android and iOS using platforms and technologies previously investigated.
This objective has been achieved successfully, achieving compliance with the scope and raised prototypes.
However, despite the fact we accomplish minimum requirements, difficulties has found in the research and implementation of the optional requirements, which might suggest that some may have been too ambitious or has partially failed in its performance.

On the issue about transforming the code to bundle a final app, the major impediment was found by Apple's hands, I was lucky to count with my project director's Mac and University of La Laguna iOS developers license, if that wasn't the case, I couldn't have generate the application for Apple's platform.

Talking about personal assessment, saying that I had to evolve and learn enough to realize this project, of which at first I did not think It could reach such complexity.
An example of this was, to my surprise, the first project's prototype, the one responsible for rendering an image took about 500 lines of code.
Even taking into account that he must build the backbone of the application, I never considered the possibility that I could take both.
Chances are it was the result of my inexperience, but it made me work harder, reinforce skills already possessed and learn new other ones about mobile game development and especially technologies used in the project: CocoonJS and JavaScript

5.2 Trabajos futuros

Los resultados obtenidos durante el transcurso del presente trabajo han sido bastante satisfactorios, no obstante, al tratarse de un prototipo, todavía está muy lejos de convertirse en una aplicación de verdad, aún queda mucho que mejorar y bastante en lo que se puede profundizar.
A continuación se muestra una lista con mejoras estructurales que se podrían realizar sobre el prototipo:

- Crear un soporte mediante bases de datos que permita al usuario descargar diferentes imágenes para crear puzzles a resolver.
- Permitir al usuario acceder a la galería del móvil para elegir la imagen que quiera de entre las que tiene para resolver el puzzle. Esta funcionalidad queda fuera del alcance de CocoonJS actualmente, lo que no lo descarta en una versión posterior.
- Permitir al usuario guardar la imagen del puzzle en su dispositivo móvil.
- Agregar elementos de configuración y sonido a la interfaz.
- Mejorar el aspecto visual.

Además de todos estos puntos, dotar al juego de una estructura multijugador es un gran punto a tener en cuenta. Dicha estructura, que se había previsto investigar, resulta que no es muy compleja. La idea radica en que un usuario pueda sacar una imagen desde la cámara de su dispositivo y enviarla como puzzle a resolver a otro usuario que se encuentre en su lista de contactos.
Para poner en marcha esta idea se tiene que contemplar lo siguiente:

- Se debe de poder acceder a la lista de contactos del dispositivo y ver quién de ellos tiene instalado el juego Puzzleyap para poder usar este modo.
- Se debe de contar con un servidor que maneje una base de datos que contenga la información de las imágenes enviadas y que se encargue de enviar los datos de la imagen al usuario final.
- Se debe de implantar un servicio de notificaciones para Puzzleyap que informe al usuario de una petición de envío de puzzle por parte de uno de sus contactos.

No obstante, aunque la estructura no sea muy compleja, se requiere de tiempo además de que su implementación no es algo trivial, por lo que debido a los requisitos en tiempo planeados para este proyecto, se ha escapado del alcance la implementación de este sistema.
Por su parte, CocoonJS no nos permite acceder a la lista de contactos del usuario, por lo que el primer requisito de nuestro modo multijugador no se cumpliría, siendo necesaria la intervención de algún otro framework o librería para realizar nuestro propósito.

A pesar de esto, CocoonJS nos provee de soporte multijugador y servicio de notificaciones. En cuanto al soporte multijugador, el requisito fundamental es autenticarse en una red social definida para llevarlo a cabo. Esto abre una nueva posibilidad que se podrá tomar en cuenta: en lugar de acceder a

los contactos del dispositivo, se accede a las personas que te sigan en twitter, por ejemplo, u otra red social. Finalmente, el servicio de notificaciones de CocoonJS cumple con nuestras expectativas para las peticiones de envío de puzzle por parte de nuestros contactos, lo cual hace que tengamos los elementos necesarios para implantar este modo.

Finalmente y tras haber barajado estas mejoras, se podría proponer, en último lugar, el estudio de la posibilidad de comercializar la aplicación final resultante, viendo los diferentes modelos de negocio y cómo encarar el mercado actual.

Future Works

The results obtained during the course of this work have been quite satisfactory, however, being a prototype, it is still far from becoming a real application, there is still quite a lot to improve and more things that can be deepened.

The list below show structural improvements that could be made on the prototype:

- Create a database support by allowing users to download different images to create puzzles to solve.
- Allow users to access mobile gallery to choose the image you want from among those you have to solve the puzzle. This functionality is outside the scope of CocoonJS currently, which does not rule in a later version.
- Allow the user to save the image of the puzzle in their own mobile device.
- Add settings elements and sound.
- Improve visual appearance.

In addition to these points, give the game a multiplayer structure is a great point to consider. This structure, which was planned to investigate, it is not very complex. The idea is that a user can take a picture from the camera of the device and send it as a puzzle to solve to another user who is in the user contact list.

To implement this idea has to consider the following:

- You should be able to access the device's contact list and see which of them has the game Puzzleyap installed to send a puzzle.
- You must have a server that manages a database containing information of images sent and to be responsible for sending the image data to the end user.
- We must implement a notification service to inform the user when there is a puzzle request from one of their contacts.

However, although the structure is not very complex, it requires time and its implementation is not trivial, that is the reason behind the requirements planned for this project time has escaped the scope to implement this system.

Meanwhile, CocoonJS doesn't provide us to access the user's contact list, we can't comply our first multiplayer's requirement, the intervention of some other framework or library must be necessary.

Despite this, CocoonJS provides us with multiplayer support and notification service. As for multiplayer support, the key requirement is to authenticate to a defined social network to carry it out. This opens a new possibility can be taken into account: for accessing the device contacts are accessible to people who follow you on twitter, for example, or another social network. Finally, the notification service CocoonJS meets our expectations for sending puzzle requests by our contacts, which means that we have the elements necessary to implement this mode.

Given these possible improvements could also study how to market the resulting final application, looking into the different business models and how to address the current market.

Referencias

[1] << ABC Tecnología - La industria del videojuego valdrá más de 100.000 millones de dólares en 2017 >> Disponible en: http://www.abc.es/tecnologia/videojuegos/20140116/abci-industria-videojuegos-milones-euros-201401161323.html

[2] << W3C - HTML5 >> Disponible en: http://www.w3.org/TR/html5/

[3] << W3C - Javascript >> Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript

[4] << W3C - CSS >> Disponible en: http://www.w3.org/TR/CSS/

[5] << Wikipedia - Jigsaw Puzzle >> Disponible en: http://en.wikipedia.org/wiki/Jigsaw_puzzle

[6] << Apple Store >> Disponible en: http://store.apple.com/es

[7] << Google Play >> Disponible en: https://play.google.com/store?hl=es

[8] << World Wide Web Consortium (W3C) >> Disponible en: http://www.w3.org/

[9] << HTML Canvas 2D Context >> Disponible en: http://www.w3.org/TR/2012/CR-2dcontext-20121217/

[10] << Wikipedia - Adobe Flash Professional >> Disponible en: http://es.wikipedia.org/wiki/Adobe_Flash

[11] << Steve Jobs - Thoughts on Flash >> Disponible en: http://www.apple.com/hotnews/thoughts-on-flash/

[12] << Tu expero IT - Facebook, HTML5 desbanca a Adobe Flash en la red social >> Disponible en: http://www.tuexpertoit.com/2010/04/29/facebook-html5-desbanca-a-adobe-flash-en-la-red-social

[13] << AnaitGames - Flash se retira del mercado móvil >> Disponible en: http://www.anaitgames.com/noticias/flash-se-retira-del-mercado-movil

[14] << Can I use - Canvas basic support >> Disponible: http://caniuse.com/canvas

[15] << Construct 2 >> Disponible en: http://www.anaitgames.com/noticias/flash-se-retira-del-mercado-movil

[16] << ImpactJS >> Disponible en: http://impactjs.com/

[17] << EaselJS >> Disponible en: http://createjs.com/#!/EaselJS

[18] << PixiJS >> Disponible en: http://www.pixijs.com/

[19] << GameMaker >> Disponible en: http://www.yoyogames.com/studio

[20] << Xataka - Android ya acapara el 80% de cuota de mercado en smartphones y la mitad son teléfonos Samsung >> Disponible en: http://www.xataka.com/moviles/android-ya-acapara-el-80-de-cuota-de-mercado-en-smartphones-y-la-mitad-son-telefonos-samsung

[21] << PhoneGap >> Disponible en: http://phonegap.com/

[22] << Appcelerator Titanium >> Disponible en: http://www.appcelerator.com/titanium/

[23] << Corona SDK >> Disponible en: http://coronalabs.com/products/corona-sdk/

[24] << CocoonJS >> Disponible en: https://www.ludei.com/cocoonjs/

[25] << CocoonJS - How to use >> Disponible en: https://www.ludei.com/cocoonjs/how-to-use/

[26] << Revolución Móvil - Tutorial: ¿Cuándo usar PhoneGap? >> Disponible en : http://revolucion.mobi/2012/10/23/cuando-usar-phonegap/

[27] << Lo-Dash >> Disponible en: http://lodash.com/

[28] << Zebkit >> Disponible en: http://www.zebkit.com/

[29] << W3Schools - HTML Canvas Reference >> Disponible en: http://www.w3schools.com/tags/ref_canvas.asp

[30] << Paper.js>> Disponible en: http://paperjs.org/

[31] << Kinetics.js >> Disponible en: http://kineticjs.com/

[32] << W3C - Device Orientation >> Disponible en: http://w3c.github.io/deviceorientation/spec-source-orientation.html

[33] << glfx.js >> Disponible en: http://evanw.github.io/glfx.js/

[34] << Geek Office Dog - Hello Swirl! >> Disponible en: http://geekofficedog.blogspot.com.es/2013/04/hello-swirl-swirl-effect-tutorial-in.html

Bibliografía consultada

[1] << HTML5 Canvas Lessons >> Disponible en: http://www.iguanademos.com/Jare/docs/html5/Lessons/

[2] << CSS Tricks - Learning Canvas: Making a Snake Game >> Disponible en: http://css-tricks.com/learn-canvas-snake-game/

[3] << Smashing Magazine - How to design a mobile game with HTML5 >> Disponible en: http://www.smashingmagazine.com/2012/10/19/design-your-own-mobile-game/

[4] << Christopher Bennage - Building a Game With JavaScript: Start Screen >> Disponible en: http://dev.bennage.com/blog/2013/01/11/game-dev-02/

[5] << CSS Deck - Ping-Pong Game Tutorial with HTML5 Canvas and Sounds >> Disponible en: http://cssdeck.com/labs/ping-pong-game-tutorial-with-html5-canvas-and-sounds

[6] << Game Dev Academy - Create a Game UI with the HTML5 CANVAS >> Disponible en: http://www.gamedevacademy.org/create-a-game-ui-with-the-html5-canvas/

[7] << Microsoft Developer Network - Modernize your HTML5 canvas game >> Disponible en: http://msdn.microsoft.com/en-us/hh969228.aspx

[8] << Stack Overflow - HTML5 Canvas Font Size Based on Canvas Size >> Disponible en: http://stackoverflow.com/questions/22943186/html5-canvas-font-size-based-on-canvas-size

[9] << Microsoft Developer Network - Cómo usar canvas, SVG y multitáctil para crear un juego de rompecabezas en mosaico >>

Disponible en: http://msdn.microsoft.com/es-es/library/jj152139(v=vs.85).aspx

[10] << Destaca - Tutorial: Diseño y maquetación web para smartphones y tablets >> Disponible en: http://www.destacaimagen.com/diseno-y-maquetacion-web-smartphones-y-tablets/

[11] << Stack Overflow - Capturing only a portion of canvas with .todataurl Javascript/HTML5 >> Disponible en: http://stackoverflow.com/questions/14017442/capturing-only-a-portion-of-canvas-with-todataurl-javascript-html5

[12] << Integralist - Building a game with HTML5 Canvas >> Disponible en: http://www.integralist.co.uk/posts/building-a-game-with-html5-canvas/

[13] << Code Project - Jigsaw Puzzle Game in HTML5 >> Disponible en: http://www.codeproject.com/Articles/366857/Jigsaw-Puzzle-Game-in-HTML

[14] << Code Project - HTML5 Jigsaw Puzzle >> Disponible en: http://www.codeproject.com/Articles/395453/Html-Jigsaw-Puzzle

[15] << Code Project - WPF Jigsaw Puzzle >> Disponible en: http://www.codeproject.com/Articles/117037/WPF-Jigsaw-Puzzle

[16] << iDiallo - Javascript Game State Stack Engine >> Disponible en: http://idiallo.com/blog/javascript-game-state-stack-engine#b

[17] << Treehouse - Create Vector Masks using the HTML5 Canvas >> Disponible en: http://blog.teamtreehouse.com/create-vector-masks-using-the-html5-canvas

[18] << Emanuele Feronato - Creation of a jigsaw puzzle using HTML5 Canvas and KineticJS – step3: jigsaw generation >> Disponible en:

http://www.emanueleferonato.com/2013/01/08/creation-of-a-jigsaw-puzzle-using-html5-canvas-and-kineticjs-step3-jigsaw-generation/

[19] << DreamDealer - Javascript performance tips and tricks >> Disponible en:

http://www.dreamdealer.nl/articles/javascript_performance_tips_and_tricks.html

[20] << HTML5 Rocks - Improving HTML5 Canvas Performance >> Disponible en:
http://www.html5rocks.com/en/tutorials/canvas/performance/

[21] << Stack Overflow - Proper way to detect WebGL support >> Disponible en: http://stackoverflow.com/questions/11871077/proper-way-to-detect-webgl-support

[22] << HTML5 Rocks - This End Up: Using Device Orientation >> Disponible en:
http://www.html5rocks.com/en/tutorials/device/orientation/

[23] << Chrome - Pixel perfect UI in the WebView >> Disponible en:
https://developer.chrome.com/multidevice/webview/pixelperfect

[24] << HTML5 Canvas Tutorials >> Disponible en:
http://www.html5canvastutorials.com/

[25] << Oscon - Chapter 3. The HTML5 Canvas Text API >> Disponible en:
http://chimera.labs.oreilly.com/books/1234000001654/ch03.html

[26] << Bitelia - 5 librerías en JavaScript para hacer juegos en HTML5 >> Disponible en: http://bitelia.com/2014/08/librerias-javascript-para-hacer-juegos

[27] << Mug - Desarrollando videojuegos con HTML5 y Javascript >> Disponible en: http://www.mug-it.org.ar/343063-Desarrollando-videojuegos-con-HTML5-y-Javascript.note.aspx

[28] << Etnassoft - Canvas y videojuegos en HTML5 >> Disponible en: http://www.etnassoft.com/2011/01/09/canvas-y-videojuegos-en-html5/

[29] << Desarrollo Web - Guía del Canvas HTML5 para desarrolladores >> Disponible en: http://www.desarrolloweb.com/articulos/guia-canvas-html5-desarrolladores.html

[30] << Mobiforge - HTML5 for the Mobile Web: Canvas >> Disponible en: http://mobiforge.com/design-development/html5-mobile-web-canvas

[31] << Esandra - 10 Soluciones para Crear Apps Multiplataforma >> Disponible en: http://www.esandra.com/soluciones-para-crear-aplicaciones-moviles-multiplataforma

[32] << Jose Marín de la Fuente - Frameworks multiplataforma basados en HTML5 >> Disponible en: http://www.marindelafuente.com.ar/frameworks-multiplataforma-basados-en-html5/

Anexo. Juegos de puzzles estudiados

[1] << Foto puzzle >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.appsdgl.photoPuzzle

[2] << Puzzle fotos >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.xuecs.PhotoPuzzle&hl=es

[3] << Puzzle con tu foto >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.puzzle.withphoto

[4] << Jigsaw Puzzle >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.criticalhitsoftware.jigsawpuzzle

Disponible en iTunes: https://itunes.apple.com/us/app/id495583717

[5] << Rompecabezas >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.titan.jigsaw

[6] << Jigsaw Puzzles >> Disponible en Google Play: https://play.google.com/store/apps/details?id=com.yodesoft.android.game.yopuzzle

[7] << Rompecabezas Jigty >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.outfit7.jigtyfree

Disponible en iTunes: https://itunes.apple.com/app/id672246969?mt=8

[8] << Jigsaw Puzzle Maker >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.puzzleworldgames.jigsawfreeplay

[9] << Real Jigsaw >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.rottzgames.realjigsaw

[10] << Jigsaw World >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.inertiasoftware.jigsawworld

[11] << Jigsaw Puzzles >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.altarsoft.jigsawpuzzles

Disponible en iTunes: https://itunes.apple.com/us/app/jigsaw-puzzles-pics/id890889131

[12] << Jigsaw Puzzle 500+ >> Disponible en iTunes:

https://itunes.apple.com/es/app/jigsaw-puzzle-500+/id521510254?mt=8

[13] << Rompecabezas! >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.M2H.jigsaw&hl=es
Disponible en iTunes:
https://itunes.apple.com/us/app/rompecabezas!/id445358235?l=es&mt=8

[14] << Puzzles y Jigsaw >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=com.rangames.puzzlesjigsaws

Disponible en iTunes: https://itunes.apple.com/es/app/puzzles-jigsaws-simulador/id523642255?mt=8

[15] << Jigsaw Puzzles Saga >> Disponible en Google Play:

https://play.google.com/store/apps/details?id=air.wms.jigsaw.puzzles.saga&hl=es

Disponible en iTunes: https://itunes.apple.com/es/app/jigsaw-puzzles-saga/id870701460?mt=8