

UNIVERSIDAD POLITÉCNICA DE MADRID

FACULTAD DE INFORMÁTICA

Trabajo de Fin de Carrera

Aplicación de apuestas para iPhone 3G

AUTOR: Francisco Miguel Merchán Casado

TUTOR: Ángel Herranz Nieva

Índice general

1. Resumen	5
2. Introducción	7
2.1. Mundo de las apuestas y BetFair	7
2.2. API de servicios de BetFair	9
2.3. Apple y su SDK de desarrollo para iPhone	10
3. Objetivos	13
4. Requisitos	15
4.1. Escenario	15
4.2. Historias de Uso vs Caso de uso	15
4.3. Historias recopiladas	15
4.3.1. Instalación	15
4.3.2. Actualizar la aplicación	15
4.3.3. Desinstalar	15
4.3.4. Ejecución	16
4.3.5. Configurar la aplicación	16
4.3.6. Gestión de los Eventos	16
4.3.7. Realizar una apuesta	16
4.3.8. Gestionar las apuestas	16
4.3.9. Realizar Trading sobre una apuesta ya realizada	16
5. Arquitectura	17
5.1. Descripción general	17
5.2. Arquitectura del iPhone y su SDK	17
5.3. Arquitectura del API de BetFair	20
5.4. Arquitectura de la aplicación	21
5.4.1. Núcleo de la aplicación	21
5.4.2. Crontroladores de Vista	22
5.4.3. Controlador Vista de tablas	24
5.4.4. Arquitectura de la aplicación	24
5.5. Diagrama de Clases	28

6. Implementación	29
6.1. Entorno de ejecución	29
6.2. Estructura de la aplicación	30
6.3. Internalización	30
6.4. Interpretación XML de BetFair a Cocoa Touch	30
7. Conclusiones	31

Capítulo 1

Resumen

En este documento se describe el trabajo de fin de carrera acerca del desarrollo de una aplicación de apuestas de BetFair para el dispositivo iPhone. Esta aplicación contiene las siguientes funcionalidades:

- Consultar los eventos de BetFair por los que se puede apostar
- Apostar en los eventos.
- Realizar seguimientos a la evolución de los mercados.
- Asesoramiento para realizar Tradings sobre apuestas ya realizadas.

A continuación se explica con más detalle la realización del trabajo.

Capítulo 2

Introducción

2.1. Mundo de las apuestas y BetFair

Actualmente las apuestas deportivas por internet tienen gran aceptación en todo el mundo. En España ya hay una tradición de bastantes años por la quiniela y la lotería. Sin embargo, actualmente ya empiezan a estar desfasadas por poca capacidad de adaptarse a tiempos y gustos nuevos. En la quiniela, al tratarse de cantidades variables nunca sabemos por anticipado cuánto cobraremos en caso de acierto. Un acierto de 13 resultados puede convertirse en una gran decepción por el premio conseguido.

Con el gran desarrollo de internet surgen las casas de apuesta en línea como Bwin.com, BetFair y MiApuesta.com, por citar las más conocidas. Las casas de apuestas nos ofrecen una gran variedad de apuestas en diferentes deportes que nos permiten adecuarnos a nuestros gustos y conocimientos. Esto unido a la evolución de Internet, facilita el acceso a la información y el pago de las apuestas a través de transferencias electrónicas seguras.

**** Para facilitar en entendimiento y la comprensión acerca de la temática de este trabajo de Fin de Carrera definiremos los principales conceptos en una apuesta:

- Bankroll: Es la cantidad de dinero (capital) que estamos dispuestos a apostar. Lógicamente tendremos que administrarlo de la mejor manera posible, minimizando las pérdidas en caso de perder una apuesta y maximizar las ganancias de una apuesta ganadora. La disciplina y el conocimiento del mercado son las mejores aliadas de nuestro Bankroll. La mayoría de las casas de apuesta por Internet nos regalan un mínimo Bankroll para animarnos a participar en ellas.
- Stake: Es la cantidad que ofrecemos en la apuesta por un determinado evento. Es decir, apostar 10? a que gana el Sevilla F.C el campeonato de liga de 2009.

- Odds: Es la cuota (probabilidad) de una apuesta. Por ejemplo se paga 3 que Fernando Alonso gane el mundial de F1
- Beneficio: Es la ganancia que obtenemos de una apuesta ganadora cuyo resultado es el siguiente: $\text{Beneficio} = (\text{stake} \times \text{odd}) - \text{stake}$
- Pida: Lógicamente lo opuesta al beneficio. Lo que todo apostante quiere evitar.
- Yield: Espresado en porcentaje expresa el beneficio obtenido del total apostado. Es lo que diferencia a un buen apostador de los demas. El cálculo es el siguiente: $\text{Yield} = (\text{Beneficio} / \text{Total apostado}) \times 100$

Un modalidad de apuesta derivada de la aparición de las casas de apuestas en Internet es el Trading. Se trata de apostar en contra de tu ultima apuesta sobre un mismo evento obteniendo asi una ganancia segura en unos casos o minimizar la pérdida en otros. La primera casa de apuestas en ofrecer este servicio fue BetFair.com. Este método se suele dar en dos escenarios típicos:

- Suceso sobrevalorado: apostamos por un evento con una cuota alta y según se va acercando al final del evento disminuye la misma.
- Suceso infravalorado: apostamos por un evento con una cuota baja y esta va subiendo según se va acercando el final de evento.

La ventaja de realizar este tipo de apuestas es que en función de los resultados que hagamos podemos obtener beneficios pase lo que pase en el evento apostado.

En el siguiente ejemplo exponemos un caso típico de Trading para su mejor entendimiento:




Villarreal v Real Madrid			Iguaradas: EUR 8.683	Actualizar
<input type="checkbox"/> En modo En Juego <input checked="" type="checkbox"/> Resultados Live				
<input checked="" type="checkbox"/> A favor/En Contra <input type="checkbox"/> Profundidad de Mercado			Más opciones ▶	
Selecciones: (3)			A favor	En contra
	Villarreal		2.28	2.36
	Real Madrid		3.25	3.35
	Empate		3.7	3.75

Figura 2.1: Situacion inicial

Tal y como observamos en la figura X, vemos que el favorito para la victoria del partido es Nadal. Su cuota a favor esta mucho más alta que la de Roger Federer. En el mundo de las apuestas, influyen todos los pequeños factores que al aficionado se les escapa. Supongamos que Nadal pierde el primer set, esto desemboca en cambios en la tabla de apuestas.




Villarreal v Real Madrid			Iguales: EUR 35.851	Actualizar
<input type="checkbox"/> En modo En Juego <input checked="" type="checkbox"/> Resultados Live				
<input checked="" type="checkbox"/> A favor/En Contra <input type="checkbox"/> Profundidad de Mercado			Más opciones ▶	
Selecciones: (3)	A favor		En contra	
 Villarreal	2.28		2.3	
 Real Madrid	3.3		3.4	
 Empate	3.75		3.8	

Figura 2.2: Situacion cambiante

Vemos ahora que el mercado ha cambiado. Ahora mismo se refleja una clara ventaja para la victoria del Roger Federer.

En este ejemplo, al principio realizamos una apuesta a favor de Rafa Nadal por 50 euros, con lo que tendríamos:

Beneficio = $(1.8 \times 50) - 50 = 40$ euros si gana Rafa Nadal Riesgo = -50 euros si pierde Nadal

Un tiempo más adelante, el mercado a cambiado debido al resultado del primer set, Nadal ya no es tan favorito para la victoria.

Para asegurar el dinero apostado anteriormente realizamos una apuesta en contra de Nadal. De tal forma que nos queda:

Beneficio = 70 euros si pierde Nadal Riesgo = $70 - (1.2 \times 70) = -14$ euros si gana Nadal

Al final tenemos que:

Si gana Rafa Nadal obtenemos : $40 \text{ euros} - 14 = 26$ euros de beneficio Si pierde Nadal y por tanto gana Roger Federer : $70 - 50 = 20$ euros de beneficio

Con lo que estamos cubierto ante cualquier resultado del partido. En ambos casos tenemos ganancias. El escenario expuesto es uno de los mejores casos que nos pueden dar ya que obtenemos beneficio en ambos casos. Pero puede suceder que hayamos apostado por un evento con demasiada confianza y luego en un futuro vemos que puede ser una ruina. En ese escenario el objetivo prioritario sería minimizar la pérdida apostando en contra del evento en cuestión.

2.2. API de servicios de BetFair

En el año 2007 BetFair es el primer portal de apuestas que ofrece un API de acceso a sus servicios. La estrategia es clara, enganchar a los desarrolladores para crear todo tipo de aplicaciones de apuestas usando su portal de apuestas como base. Como era de espera empezaron a salir multitud de aplicaciones dirigidas a un sector experto (para todos aquellos que se les queda corto el portal de internet) y aplicaciones para principiantes. Ahora, con las nuevas tecnologías móviles, el usuario quiere / necesita realizar consultas en todo momento de sus servicios contratados. Ya sea email, saldo de su cuenta del banco..... es la hora

de las aplicaciones móviles.

2.3. Apple y su SDK de desarrollo para iPhone

Apple inició la revolución del ordenador personal en la década de los setenta con el ordenador Apple II y reinventó el ordenador personal en los ochenta con el Macintosh. Hoy, Apple sigue liderando la industria en innovación con sus premiados ordenadores de sobremesa y portátiles, el sistema operativo OS X, iLife (aplicación de creación de contenido multimedia) y sus aplicaciones profesionales. Apple está también en la vanguardia de la revolución musical con sus reproductores de música y vídeo portátiles iPod y la tienda de música online iTunes, e irrumpió en 2007 en el mercado de la telefonía móvil con su revolucionario iPhone.



Figura 2.3: iPhone 3G

El iPhone muy pronto se convirtió en un fenómeno extraordinario. Horas antes de su salida al mercado ya había colas de espera de más de 8 horas, nunca antes visto en un lanzamiento de un móvil. Su gran pantalla unida a las más novedosas tecnologías del momento convirtieron al dispositivo en alcanzar el millón de ventas en EEUU. Pero no todo fueron buenas noticias para Apple. En Europa, donde no se vendía el dispositivo, criticaban el dispositivo por su apuesta en tecnología obsoleta en Europa, GPRS (acceso similar al bando de ancha de un modem analógico) para acceder a las redes de datos de telefonía y su escaso soporte para crear contenido para el dispositivo.

Por todo esto, en Marzo de 2008 Apple comunica su deseo de lanzar su tienda de aplicaciones para el iPhone para Julio del mismo año y lanza un conjunto de herramientas de desarrollo (SDK) para todos aquellos desarrolladores que quieran participar en la misma.

En Europa hubo que esperar hasta Julio de 2008, de la mano de Telefónica se lanza el iPhone 3G, una versión mejorada del iPhone con tecnología de acceso a red 3G. Con este avance el dispositivo se afianza como uno de los más rápidos para acceder a la red de redes, Internet, y hacer uso de los servicios web 2.0 (fotos, blogs, localización, voz por IP...).

Hoy en día, se ha alcanzado el millón de descargas de aplicaciones en el iTunes App Store (Tienda de aplicaciones del iPhone) confirmando el éxito del dispositivo.

He aquí la importancia de la temática de este Trabajo de FIn de Carrera, la adaptación del API de Betfair a la plataforma iPhone y la creación de una aplicación para hacer uso de los servicios de esta casa de apuestas.

Capítulo 3

Objetivos

Los objetivos que se pretendan cubrir con el actual trabajo de fin de carrera son los siguientes:

- Posibilidad de apostar por eventos del portal BetFair.com usando su API de acceso a sus servicios web.
- Asesorar el usuario sobre cuando realizar el trading a las apuestas ya realizadas.

Capítulo 4

Requisitos

4.1. Escenario

El escenario que pretende cubrir la aplicación es la gestión de los eventos del portal de apuestas BetFair. Se pretende cubrir las mismas funcionalidades que ofrece BetFair en su portal web.

4.2. Historias de Uso vs Caso de uso

4.3. Historias recopiladas

4.3.1. Instalación

Para poder instalar la aplicación solo se necesitará una cuenta del programa iTunes Store de Apple. Es gratuito. La aplicación estará disponible dentro del programa App Store del dispositivo. Solamente habrá que descargar la aplicación de dicho portal.

4.3.2. Actualizar la aplicación

La aplicación App Store del dispositivo será la encargada de comunicar al usuario la aparición de una nueva versión del aplicativo. Para actualizarla simplemente habrá que seguir las indicaciones de dicho programa.

4.3.3. Desinstalar

Para desinstalar la aplicación simplemente se mantiene pulsado el icono de la misma una segundos e inmediatamente pulsamos sobre la X que aparecerá sobre la misma.

4.3.4. Ejecución

Para lanzar la aplicación solo hay que pulsar el icono que aparece en la pantalla principal del dispositivo.

4.3.5. Configurar la aplicación

Para poder usar los servicios de BetFair a través de la aplicación hay que configurar los datos de acceso al servicio. Para ello, hay que pulsar sobre el icono de "Ajustes" del dispositivo. Una vez abierta la pantalla buscamos la celda que contiene el nombre de la aplicación e introducimos los datos de acceso al servicio web BetFair

4.3.6. Gestión de los Eventos

Se podrá navegar y obtener información de todos los eventos disponibles del portal BetFair.

4.3.7. Realizar una apuesta

Una vez lanzada la aplicación pulsamos sobre Eventos. Navegamos por los menús hasta llegar al evento de nuestro interés. Una vez en la pantalla de información del evento observamos las cuotas y cantidades del mercado e introducimos el tipo de apuesta, la cantidad y la cuota deseados. Seguidamente pulsamos sobre el botón Apostar y confirmamos la apuesta. El sistema nos mostrará si la apuesta se ha realizado con éxito.

4.3.8. Gestionar las apuestas

La aplicación será capaz de recopilar todas las apuestas realizadas sobre BetFair. Por cada apuesta el sistema mostrará las opciones disponibles.

4.3.9. Realizar Trading sobre una apuesta ya realizada

El sistema será capaz de asesorar para realizar Trading sobre una apuesta ya realizada anteriormente. Para ello se mostrará una tabla con las opciones disponibles dentro del resumen de una apuesta ya realizada.

Capítulo 5

Arquitectura

En este capítulo se presentará la arquitectura de la aplicación. Para un mejor entendimiento de la misma explicaremos brevemente la arquitectura del dispositivo iPhone (Cocoa Touch). Seguiremos por una breve descripción del API de Betfair y terminaremos con una explicación detallada de la arquitectura desplegada en la aplicación.

5.1. Descripción general

Como ya hemos visto en capítulos anteriores los requisitos de la aplicación dan como resultado la arquitectura actual de la aplicación. Para llevar a cabo su desarrollo, hemos tenido en cuenta la arquitectura del dispositivo y las herramientas adjuntas. Todas las decisiones tomadas en la construcción de la arquitectura de la aplicación se han basado en los límites o restricciones impuestas por el uso de la arquitectura de Apple y los límites impuestos en el uso del API de Betfair. Para comprender mejor la composición de la aplicación explicaremos brevemente la arquitectura impuesta por Apple en el iPhone y la tecnología en la que se basa el API que ofrece BetFair para el uso de sus servicios.

5.2. Arquitectura del iPhone y su SDK

La Arquitectura del iPhone se basa en su sistema operativo iPhone OS capaz de ejecutar tanto aplicaciones web como aplicaciones nativas. La arquitectura del iPhone OS es similar a la arquitectura básica encontrada en los Mac OS X (ordenadores personales de Apple). Esta decisión fue tomada por Apple para facilitar la transición de los programadores de Mac a la plataforma del iPhone. En una descripción de alto nivel, el iPhone OS es un mero intermediario entre el hardware del dispositivo y las aplicaciones que disponemos en el dispositivo.

La implementación de las tecnologías del iPhone OS se puede ver como un conjunto de capas. En la parte baja del sistema se encuentran las capas encargadas de los servicios fundamentales que permiten la ejecución de las aplicaciones,

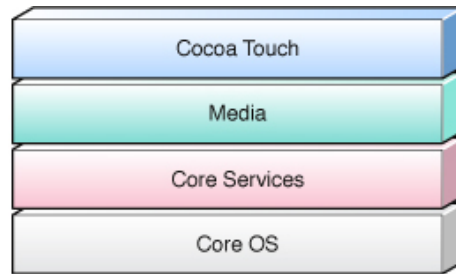


Figura 5.1: iPhoneOS system layers

y en las más altas contienen las capas sobre los servicios multimedia y de las más latas tecnologías.

- **Capa Cocoa Touch:** Es una de las capas más importantes del iPhone OS. Es la encargada de proveer la infraestructura necesaria para la implementación de aplicaciones. Contiene los siguientes frameworks:
 - **UIKit Framework:** Contiene todas las clases necesarias para la programación de la interfaz de usuario de las aplicaciones así como el acceso a las principales tecnologías físicas propias del dispositivo móvil (acelerómetro, cámara)
 - **Foundation Framework:** Es un framework heredado de la plataforma OS X. Provee el soporte para las siguientes características:
 - Colecciones de datos (arrays, sets....)
 - Gestión del Tiempo y Fechas.
 - Gestión de las preferencias del dispositivo.
 - Gestión de URLs
 - Internalizaciones del dispositivo.
 - **Address Book UI Framework:** Una de las partes más importantes en un dispositivo móvil es la agenda de contactos del dispositivo. Este framework es el encargado de gestionar toda la interfaz de usuario relacionado con los contactos del dispositivo.
- **Capa Media:**

Es la capa encargada de dar soporte a las tecnologías de gráficos, audio y video para proporcionar al usuario la mejor experiencia multimedia vista en un dispositivo móvil. Más importantes que las tecnologías en sí, esta capa fue diseñada para facilitar al desarrollador su uso para crear aplicaciones con una gran apariencia en su interfaz de usuario y una reproducción de audio genial.

A continuación enumeramos las tecnologías soportadas:

- Gráficos: Quartz, Quartz Core Animation, Open GL
 - Sonidos: ACC, Apple Lossless, A-law, IMA4, Linear PCM, u-law
 - Video: MPEG-4, H.264
- Capa Core Services: La Core Services Layer es la encargada de proporcionar los servicios básicos del sistema a las aplicaciones para su uso. Contiene los siguientes frameworks:
- AddressBook: Provee los servicios básicos para acceder a los contactos almacenados en el dispositivo.
 - Core Foundation: Es un conjunto de interfaces para dar soporte de gestión de datos y servicios para el uso de las aplicaciones del dispositivo.
 - CF Network: Conjunto de interfaces para que las aplicaciones hagan uso de los protocolos de red implementados en el terminal.
 - BSD Sockets.
 - Conexiones encriptadas
 - Resolución de nombres DNS
 - HTTP, HTTPS
 - Servicios Bonjour
 - Core Location: Framework que permite determinar la actual posición (latitud, longitud) del dispositivo. Hace uso del dispositivo GPS implementado en el dispositivo. Ofrece todo lo necesario para que las aplicaciones implementen características de localización o guiado.
 - Security: Garantiza el acceso seguro a los datos y aplicaciones almacenadas en el dispositivo.
 - SQLite: Soporte para gestionar bases de datos para que las aplicaciones gestionen sus datos.
 - Soporte a XML: Soporte para dar a las aplicaciones métodos para manipular contenido XML.
- Capa Core OS:

Es la capa que contiene el entorno del núcleo de sistema (kernel), controladores para dispositivos, interfaces básicas del sistema operativo. El kernel es el responsable de todos los aspectos relacionados con el sistema operativo. Es el encargado de gestionar la memoria virtual, threads (procesos ligeros), sistema de ficheros, red y procesos de comunicación. Los controladores son los encargados de proporcionar la interfaz entre el hardware y los frameworks de las capas superiores.

5.3. Arquitectura del API de BetFair

El API de Betfair esta diseñado bajo una interfaz SOAP que esta disponible a través de una conexión web segura. (Explicación de SOAP).

El API esta disponible a partir de un fichero en formato WSDL. Dicho formato describe la interfaz para los servicios web SOAP. Contiene todas las llamadas a los procedimientos remotos de los servicios proporcionados por BetFair. Los servicios proporcionados por el API se dividen en dos conjuntos: Global y Exchange.

El conjunto Global contiene todas las llamadas básicas referentes a los servicios de (login) a BetFair, la administración de tu cuenta BetFair, tus fondos y las llamadas para navegar por los diferentes eventos disponibles en el portal de apuestas.

El conjunto Exchange contiene las llamadas a los servicios de apuestas, es decir, apostar por un evento, descripción de los mercados disponibles, actualización o cancelación de las apuestas ya realizadas, historial de todas nuestra apuestas.....

Existen dos formas de acceso al API:

- Free Access API: con este acceso tendremos disponibles los servicios básicos del portal. Las herramientas suficientes para poder apostar por los eventos disponibles. Gratuita pero con acceso limitado a las llamadas de los servicios.
- Full Access API: Acceso de pago donde están disponibles servicios exclusivos del portal tales como información avanzada de los mercados, gestión avanzada de nuestra cuenta de usuario.... Cuota anual y sin límites en las llamadas.

Para nuestra aplicación, al no soportar los archivos WSDL, hubo que implementar cada llamada a los servicios de betfair bajo la tecnología Apple y su posterior serialización para el tratamiento de datos de la aplicación.

Los servicios básicos que hemos usado para el desarrollo de la aplicación han sido:

- Login: Llamada para poder usar los servicios web de BetFair.
- GetActiveEventTypes: Con esta llamada obtenemos todos los eventos deportivos actualmente en marcha por los que se puede apostar.
- GetAllMarkets: Con esta llamada obtenemos todos los mercados referentes a un evento.
- GetCurrentBets: Llamada por la cual obtenemos todas las apuestas activas que hemos realizado.
- GetMarketPrices: Obtenemos los precios (back y lay) actuales por un evento determinado.

- GetMarket: Esta llamada nos devuelve todos los datos referentes a un mercado.
- PlaceBets: Con esta llamada enviamos una apuesta a betfair.

5.4. Arquitectura de la aplicación

Para el diseño de la arquitectura nos hemos basado en el patrón de diseño "Modelo Vista Controlador". La razón del uso de este patrón se ha debido a dos cuestiones fundamentales:

- La facilidad entre la comunicación entre el modelo de datos y la interfaz. Añadir también que cualquier cambio en la interfaz de usuario no afecta al resto del modelo, por lo que se gana en facilidad a la hora de seguir desarrollando la solución en el futuro.
- Apple, en el uso de las herramientas de desarrollo del SDK, te recomienda el uso de dicho patrón para la creación de la interfaz de usuario de la aplicación. Esta recomendación es debido a la arquitectura interna del iPhone OS visto en un capítulo anterior y a que sus herramientas de desarrollo también están orientadas a esta solución.
- La estructura jerárquica de los datos obtenidos a través del API de Betfair: Eventos -¿Mercados -¿Runners.

5.4.1. Núcleo de la aplicación

Toda aplicación para el iPhone está desarrollada usando el framework UIKit y tienen la misma arquitectura base. UIKit proporciona todo lo necesario para lanzar la aplicación, coordinar los inputs del usuario y mostrar contenido en la pantalla.

Desde que el usuario pulsa el icono de la aplicación hasta que esta es ejecutada, el framework UIKit gestiona todo lo necesario para lanzar la infraestructura. Toda aplicación, principalmente, recibe eventos continuamente desde el sistema y debe responder a todos estos eventos. La recepción de los eventos es responsabilidad del objeto UIApplication pero la respuesta a cada uno de ellos es responsabilidad del código de la aplicación que estamos desarrollando.

El ciclo de vida de una aplicación constituye la secuencia de eventos que ocurren entre la ejecución y la finalización de la aplicación. En el SO del iPhone el usuario lanza la aplicación pulsando sobre el icono de la misma. En un corto periodo de tiempo, el sistema muestra una animación y procede a la ejecución de la aplicación en cuestión llamando a la función "main". A partir de este momento, UIKit es el encargado de lanzar la interfaz de usuario y de leer los eventos que se produzcan en un bucle hasta que la aplicación sea finalizada o bien por el sistema o bien por el usuario. Durante el bucle, UIKit coordina la llegada de eventos a determinados objetos que determinemos en nuestra aplicación así como las coordinar las respuestas de las mismas.

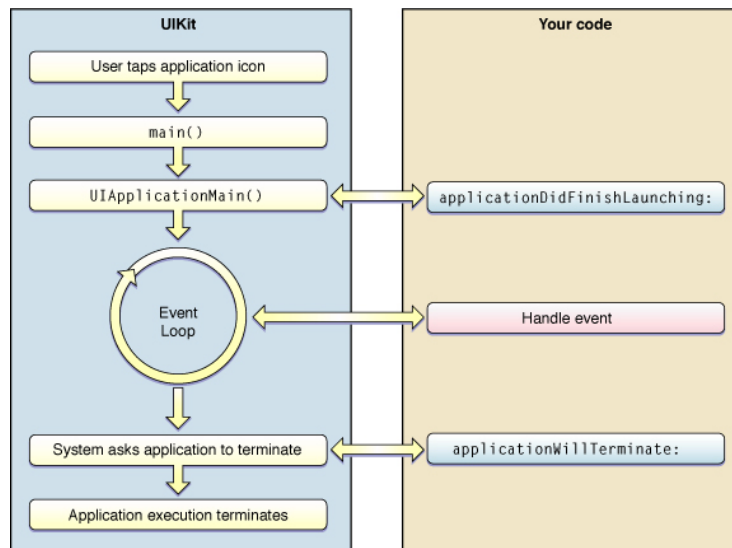


Figura 5.2: Ciclo de vida de una aplicación iPhone

La figura muestra el ciclo de vida de una aplicación para el iPhone. Tal y como vemos en la figura, UIKit se encarga del arranque de la aplicación y de la gestión de eventos. Nuestro código será el encargado de gestionar esos eventos justo cuando reciba la notificación de que la aplicación ha sido lanzada. También podremos gestionar las acciones oportunas (guardar preferencias o cambios producidos en la aplicación) cuando UIKit nos notifique de la finalización de la aplicación.

En el iPhone solo se finaliza una aplicación por tres motivos:

- El usuario desea finalizar la misma pulsando el botón "Home".
- El sistema recibe una notificación prioritaria que atender (una llamada por ejemplo) y por tanto finaliza la aplicación.
- El sistema detecta un comportamiento erróneo de la aplicación y para salvaguardar la estabilidad del sistema finaliza nuestra aplicación.

5.4.2. Controladores de Vista

Un controlador de vista proporciona la lógica básica de interfaz de usuario para dibujar las vistas de la aplicación al usuario. Apple dispone de varios patrones de interfaces de usuario para ayudar a representar en las aplicaciones el conjunto de datos hacia el usuario dentro de los dispositivos móviles.

Un controlador de vistas gestiona la vista de nuestra aplicación que aparecen entre las barras superiores e inferiores (véase figura). La vista de la aplicación aparece entre la barra de estado y la barra de navegación si ésta está presente.

Análogamente, también aparece entre la barra de esta y la barra de tabuladores o la barra de herramientas. En una vista de aplicación se muestra una porción de datos y controles que el desarrollador quiere mostrar al usuario en un tiempo determinado. El controlador de la vista simplemente gestiona la presentación de esta vista y la siguiente en aparecer para un patrón de diseño establecido.

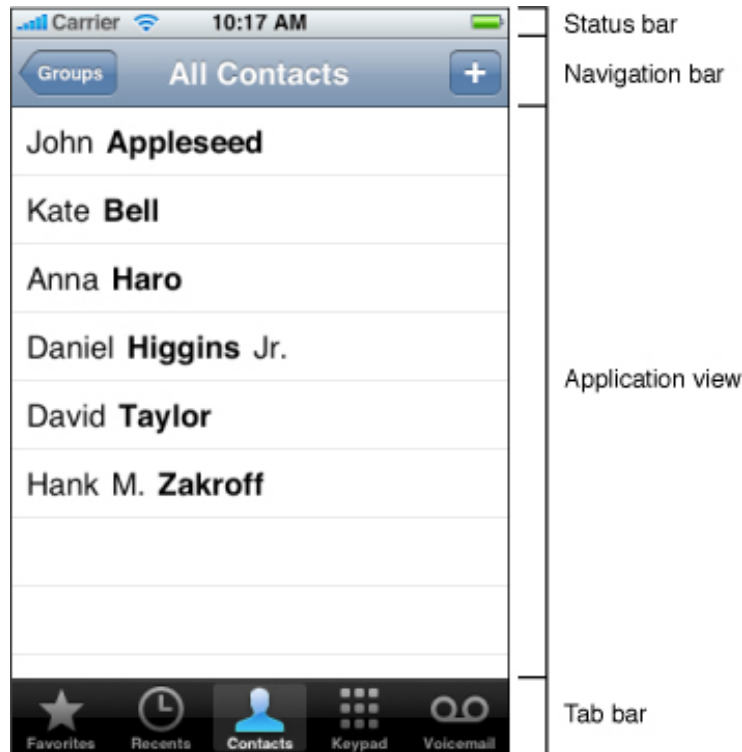


Figura 5.3: Esquema de las vistas

Usando controladores de vista eliminamos el código redundante e innecesario en las aplicaciones y proporcionamos una interfaz de usuario familiar hacia los usuarios. No necesitamos implementar la presentación de aquellas vistas de aplicaciones. También ayudan al diseño orientado de objetos separando los detalles de la interfaz de usuario de la lógica de la aplicación.

Los controladores de vista soportan el patrón de diseño Modelo - Vista - Controlador, patrón muy utilizado en la programación orientada a objetos, donde los objetos del modelo representan datos de la aplicación y son persistentes. Todos los datos y la lógica de la aplicación se ha implementado usando objetos para una mayor facilidad a la hora de la implementación. En este caso los controladores de vista son los encargados de proporcionar los métodos delegados y la fuente de datos para las vistas de tipo tabla.

5.4.3. Controlador Vista de tablas

Es muy común usar vistas de tabla para mostrar un conjunto de datos de tipo jerárquico unido a los controladores de navegación. Para ello Apple dispone de una plantilla de controlador llamada “Controlador de vistas de tabla” que nos proporciona todo lo necesario para ello. En nuestra aplicación, cuando hacemos una petición para obtener todos los eventos y mercados de Betfair este nos envía una colección de datos en forma de diccionarios y array conservando una jerarquía de eventos entre ellos. La mejor forma de representarlos es mediante una vista de tabla tal y como nos recomienda Apple. Es la siguiente figura podemos ver un esquema representativo.

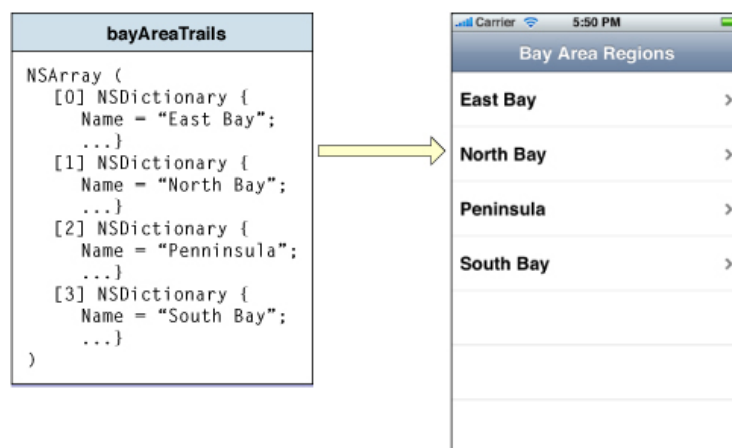


Figura 5.4: Vista de tabla del top de la jerarquía

5.4.4. Arquitectura de la aplicación

Como ya hemos visto en apartados anteriores, la aplicación esta formada principalmente por un conjunto de controladores de vista más un módulo encargado de comunicarse con el API de Betfair.

En la figura X podemos ver un esquema simplificado de la misma. Los controladores de vista se encargan de recoger los inputs del usuario. En el caso de que el input involucre datos de BetFair, el controlador envía los datos necesarios para que el módulo de comunicación obtenga la respuesta por parte del API de Betfair.

Controlador principal

Es la clase principal de la aplicación. Es la encargada de inicializar el controlador de vista que será encargado de recoger los eventos del usuario, así como de la gestión de memoria y de la gestión de los eventos del sistema.

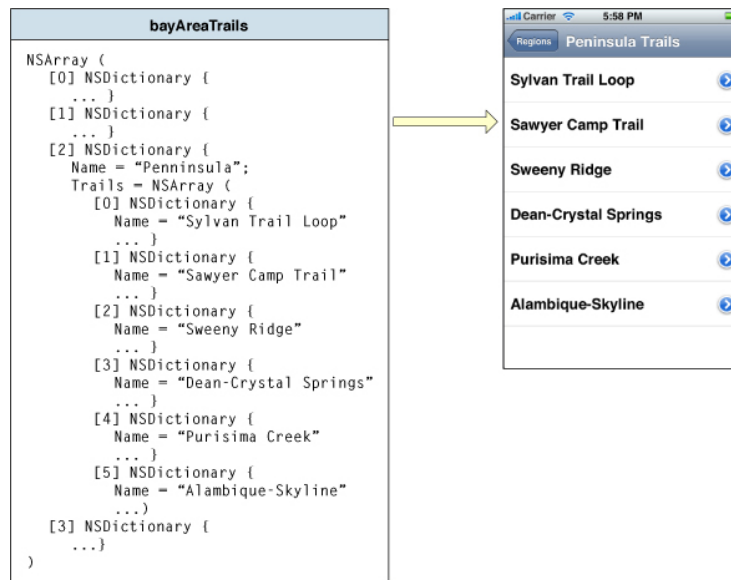


Figura 5.5: Vista de tabla de la mitad de la jerarquía

Controlador de menú principal

Esta clase es la encargada de mostrar las opciones principales del programa y recoger los inputs del usuario para lanzar el controlador adecuado a la elección del usuario. También se encarga de gestionar el login de usuario para los servicios web de Betfair. Para ello hace uso de los siguientes servicios web de Betfair a través de la clase de comunicación llamada API:

- Login: Método necesario para logarse en los servicios de Betfair y obtener un token válido de servicio.

Controlador de tipos de eventos

Es el encargado de mostrar al usuario una lista con todos los tipos de eventos activos por los que se puede apostar dentro de BetFair. En un futuro se podrá gestionar por fecha de finalización, orden alfabético o búsqueda directa de eventos. Una vez seleccionado el evento deseado lanza el controlador de eventos y mercados relacionados con el evento seleccionado. Hace uso del servicio de BetFair `GetActiveEventTypes` a través de la clase API del modelo de la aplicación.

Controlador de eventos y mercados

Es la clase encargada de mostrar de forma jerárquica los subeventos y mercados relacionados con un evento seleccionado previamente en el controlador

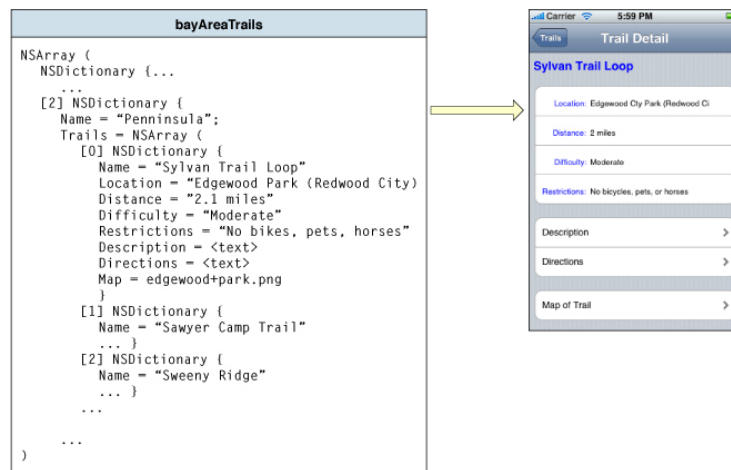


Figura 5.6: Vista de tabla detallada del final de la jerarquía

de eventos. Igualmente se podrá gestionara por fecha de finalización, orden alfabético o búsqueda directa de eventos. En caso de ser seleccionado un eventos se le mostrará los mercados y subeventos relacionados con los mismos. En caso de ser seleccionado un mercado se procederá a lanzar el controlador de información de mercado. Para ello recaba la información obtenida a través del servicio web GetEvents de BetFair.

Controlador de información de mercado

Esta clase es la responsable de gestionar toda la información relevante al mercado en cuestión. Gestionará el estado del mercado y sus propiedades obteniendo todo lo necesario del portal de Betfair a través de API de sus servicios web para ello hace uso de las llamadas del API:

- GetMarket: obtiene todos los parámetros acerca del mercado.
- GetMarketPrices: obtiene todos los precios actuales del mercado en cuestión.

Controlador de apuesta por un mercado

Es la encarga de gestionar todos los datos necesarios e introducidos por el usuarios para realizar una apuesta y enviarla a BetFair. El método usado para enviar la apuesta a BetFair a través de la clase interfaz de comunicación es PlaceBets.

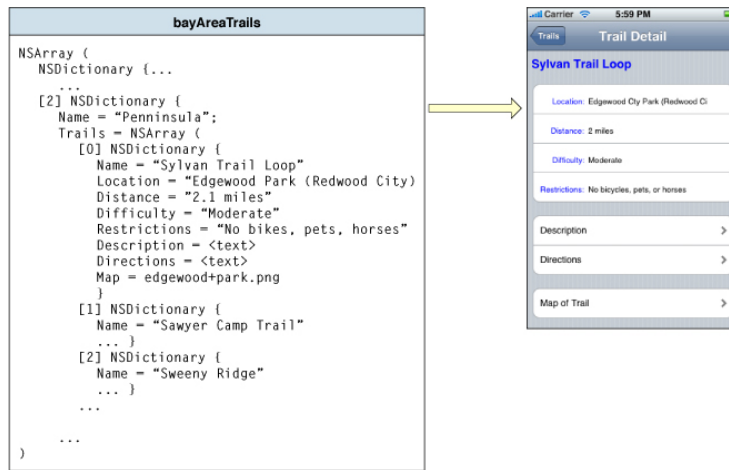


Figura 5.7: Vista de tabla detallada del final de la jerarquía

Controlador de las categorías de Mis Apuestas

Es la clase encargada de gestionar las apuestas ya realizadas en BetFair y que siguen activas. Las clasifica por categorías de mercado. Recoge la información como resultado de las llamadas a los siguientes servicios web de BetFair:

- GetMarket: obtiene todos los parámetros acerca del mercado de una apuesta en cuestión.
- GetCurrentBets: obtiene todas las apuestas realizadas por el usuario del servicios BetFair.

Controlador de Mis Apuestas

Esta clase se encarga de gestionar las apuestas realizadas por el usuario dentro de una categoría determinada. En ella se muestra resumidamente el nombre de la apuesta en cuestión, la cuota apostada y la cuota actualizada en ese momento.

Controlador de detalles de una apuesta

Se encarga de gestionar todos los detalles sobre una apuesta determinada: parámetros, stake, apuesta a favor o en contra relacionadas con la presentada... También incluye información de mercado actual completa referente al mercado de la apuesta y el trading acerca de la misma.

API: Interfaz a Betfair

Esta clase se encarga de gestionar las comunicaciones con los servidores de BetFair. Envía las peticiones a los servicios web de BetFair y se encarga

de parsear la respuesta a dichas peticiones en un formato adecuado para la aplicación.

5.5. Diagrama de Clases

Capítulo 6

Implementación

En este capítulo se describirán todos los detalles relativos a la implementación de la aplicación. Empezaremos describiendo el entorno de ejecución y del lenguaje de programación usado para su desarrollo y terminaremos destacando los detalles relevante de la implementación.

6.1. Entorno de ejecución

La aplicación desarrollada se ejecuta en un entorno llamado Cocoa Touch. Cocoa es un conjunto de frameworks orientados a objetos que proporcionan un entorno de ejecución para las aplicaciones que se ejecutan en el SO del iPhone. También forma parte del entorno de desarrollo que facilita a los desarrolladores la tarea de pasar de la etapa diseño a la etapa del desarrollo para crear una aplicación. Cocoa Touch proviene del entorno de la plataforma Mac OS. Se puede decir que es la misma plataforma añadiendo el soporte para los eventos táctiles y orientados a la tecnología móvil.

Como ya hemos comentado, Cocoa presenta dos caras: tiene una parte de entorno de ejecución y otra de desarrollo. En cuando al entorno de ejecución las aplicaciones Cocoa presentan la interfaz de usuario y están fuertemente integradas con otras partes de sistema operativo como por ejemplo el buscador del sistema de ficheros. La parte del desarrollo, Cocoa es una suite de componentes software orientados a objetos que te permiten rápidamente crear robustas y completas aplicaciones para el sistema operativo.

A pesar de que la estructura del iPhone OS es similar a la del Mac OSX, existen diferencias significantes. El diagrama del iPhone S muestra una plataforma como una serie de capas que van desde en nucleo de SSOO hasta un conjunto de framework de aplicación, la más crítica (para las aplicaciones) empieza con la capa UIKit.

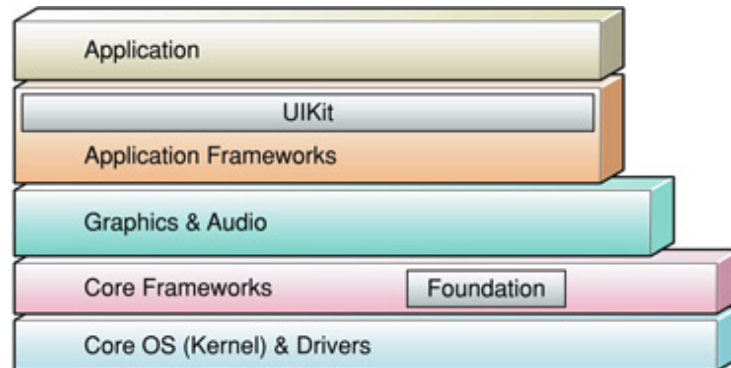


Figura 6.1: Entorno iPhone OS

6.2. Estructura de la aplicación

Cuando construimos una aplicación del iphone, Xcode lo resume en un paquete. Este paquete es un directorio en el sistema de ficheros que agrupa los recursos necesarios para la aplicación en un mismo lugar. Dicho paquete contiene el ejecutable y cualquier recurso usado por la misma (por ejemplo, el icono de la aplicación, imágenes contenidas ...). La tabla X.X lista el contenido del paquete de esta aplicación.

Insertar tablas en latex ?

Archivo	Descripción
BetFairApp	he executable file containing your application?s code. The name of this file is the same as your application name minus the .app extension. This file is required.
Eva	Dirse

6.3. Internalización

6.4. Interpretación XML de BetFair a Cocoa Touch

Capítulo 7

Conclusiones