REGISBET

Memòria Proyecto Final de Carrera– Aplicación Android

Luis José Carrasco Camacho

Consultores: Jordi Almirall López y Marc Domingo Prieto

UOC - 2º semestre 2013 - *08/01/2014*

Quiero dar mis más sinceras muestras de agradecimiento a mi familia y sobretodo a mi pareja Piedad, por la enorme paciencia y los tan bien preciados momentos de apoyo que me ha proporcionado.

Índice

Descripción del Proyecto

El porqué de esta aplicación surgió en una reunión con los amigos.

Después de una acalorada conversación surgió la típica frase "Que te apuestas si…", me di cuenta que no había forma de inmortalizar tal momento, que las palabras se las llevaba el viento y que donde se dice Diego después se puede decir Rodrigo.

Así que con el único interés de poder plasmar tal evento, de poder buscar responsabilidades cuando toque e intentar minimizar la posibilidad de hacer trampas he decidido crear esta aplicación.

La aplicación permitirá registrar mediante formularios la apuesta con un título, descripción, fecha de finalización de la apuesta, alguna imagen si se desea complementar y las posibles respuestas.

Objetivo del proyecto

El objetivo de los Trabajos de Final de Carrera (TFC) es consolidar los conocimientos adquiridos por los estudiantes de la carrera cursada.

Al no tener experiencia en el desarrollo de aplicaciones en Android el objetivo principal de este proyecto es adquirir experiencia en la planificación, el diseño, la construcción y el perfeccionamiento de la creación de aplicaciones en el Sistema Operativo Android desde cero.

La aplicación tendrá como objetivo el poder realizar un contrato virtual.

Dicho contrato lo pondrán realizar entre diferentes personas, donde estando todos de acuerdo se establece un documento contractual o apuesta base.

Esta apuesta se puede documentar gráficamente, se podrán añadir imágenes, vídeos, sonidos o textos para otorgar un apartado más gráfico e interactivo.

Se establecerá una fecha de inicio y/o una fecha de fin.

Se dará por empezada la cuenta atrás cuando todos los componentes de la apuesta o contrato dependiendo de lo riguroso que seamos, firmen un documento redactado por la figura de árbitro que puede ser uno o varios usuarios de la aplicación.

Pasado ese período de tiempo o llegado a la fecha límite se dará por finalizada la apuesta.

El creador, asignado al inicio de la apuesta, será la figura que tendrá el poder de dictaminar si se ha finalizado correctamente o no la apuesta.

Si el grupo o persona implicada ha cumplido satisfactoriamente con el objetivo marcado dará por válida la apuesta y la aplicación, previa configuración, mostrará el pacto al cual se llegó en su momento y mostrará el resultado de la apuesta.

Objetivos del TFC

El objetivo es desarrollar una aplicación real para Android que, si tiene suficiente calidad, se podría publicar en el Android Market como aplicación gratuita. Para llevarlo a cabo se tiene que poner en práctica todo lo aprendido en la carrera, entre otras cosas:

- Elaboración de un Plan de Trabajo.
- Examinar i analizar la utilización de la aplicación para realizar diseño funcional.
- Realizar de todas las funcionalidades estimadas, con calidad. Para ello se debe adquirir conocimientos en el desarrollo de software.
- Alcanzar las habilidades de gestión y planificación de un proyecto de software.
- Implementar el desarrollo de la aplicación.

- Finalmente conseguir un software útil y operativo.

Funcionalidades de RegisBet

La finalidad de este apartado es el de comunicar cuales son las funcionalidades que otorga la aplicación. Entre ellas están:

Creación apuesta

Los campos mínimos para realizar en la apuesta serán:

- Redacción de la apuesta.

- Hacia quién va dirigido, las personas podrán elegir los contactos del "agenda.

-Tipo de apuesta (pública/privada).

- Fecha de finalización, opcional.

- Insertar información gráfica (imagen), este apartado es opcional.

- **Consulta de apuesta**

  Las apuestas serán públicas, privadas o finalizadas, en caso de que tengan privilegios de consulta podrán escribir en el chat pero no podrán realizar ningún cambio en el "apuesta.

  En las apuestas públicas todos pueden acceder. En las privadas se necesita previa invitación, solamente las podrán ver aquellas personas que estén puestas en el contra.

- ***Apuesta Pública***

  Todas las personas de la aplicación podrán acceder a las apuestas públicas de los amigos.

  Si se quiere también se puede acceder a todas las apuestas vigentes de la aplicación. Se podrá ver cuál es el estado de la apuesta.

- ***Apuesta Privada***

  Solo se podrá acceder al "apuesta privada previo reto de la persona que crea la apuesta.

  Podrá ver el estado de la apuesta. Parte no desarrollada.

Login

Este apartado será el más transparente para el "usuario ya que no existirá.

Solo se "deberá bajar Regisbet para poder empezar a hacer apuestas con sus contactos de la agenda.

Configuración Perfil

Es una acción no obligatoria que modifica el perfil básico.

El usuario se crea por primera vez al entrar en la aplicación pero no tiene ningún dato guardado.

Este apartado permite guardar y actualizar los datos como nombre, fotografía o descripción.

Hardware

El obsoleto y nada práctico ordenador que utilizaba era un Acer, tenia 2 Gigas de RAM y un procesador Core Duo de 4 años. Con la instalación de la BBDD y el apache como servidores diversos podían pasar alrededor de 10 minutos hasta que encendía el ordenador y lo tenía listo para empezar a trabajar.

También cualquier cambio que realizaba era un suplicio, pues pasaba más de un minuto compilando el nuevo código.

Por ello finalmente se ha cambiado por un portátil Dell Core i5 con 8 Gigas de RAM, 300Gb de disco duro, mucho más potente y práctico para poder realizar el desarrollo de forma más cómoda.

También se ha adquirido un terminal Android Sony Ericsson Experia Hd con un núcleo con la versión 4.0.2, con el que realizar las pruebas de forma más rápida.

Software

Para el desarrollo de la aplicación se han utilizado el siguiente software:

- Sistema Operativo Windows 7.
- Eclipse con SDK de Android.
- Microsoft Sql Server 2012 Express para BBDD.
- Apache Tomcat 6.0.32 para servidor de páginas Web.
- DropBox para la gestión de Backup.
- Windroy como emulador de Android, solo al principio, después fue substituido por el emulador de Android de Eclipse.
- GanttProject para la realización del diagrama de Gantt.
- Word para realizar las memorias y diferentes entregas.
- Power Point para la presentación de TFC.
- Windows Video Maker para el montaje del vídeo.

Riesgos

El riesgo que más me preocupa es que tengo un conocimiento demasiado básico sobre el entorno de Android.

El móvil que poseo es una BlackBerry pero por suerte se ha conseguido un dispositivo que utiliza un S.O. Android 4.0.2.

El tener toda la memoria a buen recaudo siempre es necesario y un tema muy importante, para ello tengo un sistema DropBox sincronizado con otro ordenador y un viejo pen Drive de 4 Gigas para ir realizando Back-ups periódicos.

El gran volumen de trabajo que existe en mi empresa hace que sea un riesgo más a tener en cuenta ya que solo me permitirá realizar un horario siendo benevolentes de 19h a 22h los días de cada día y por suerte algunos fines de semana libres.

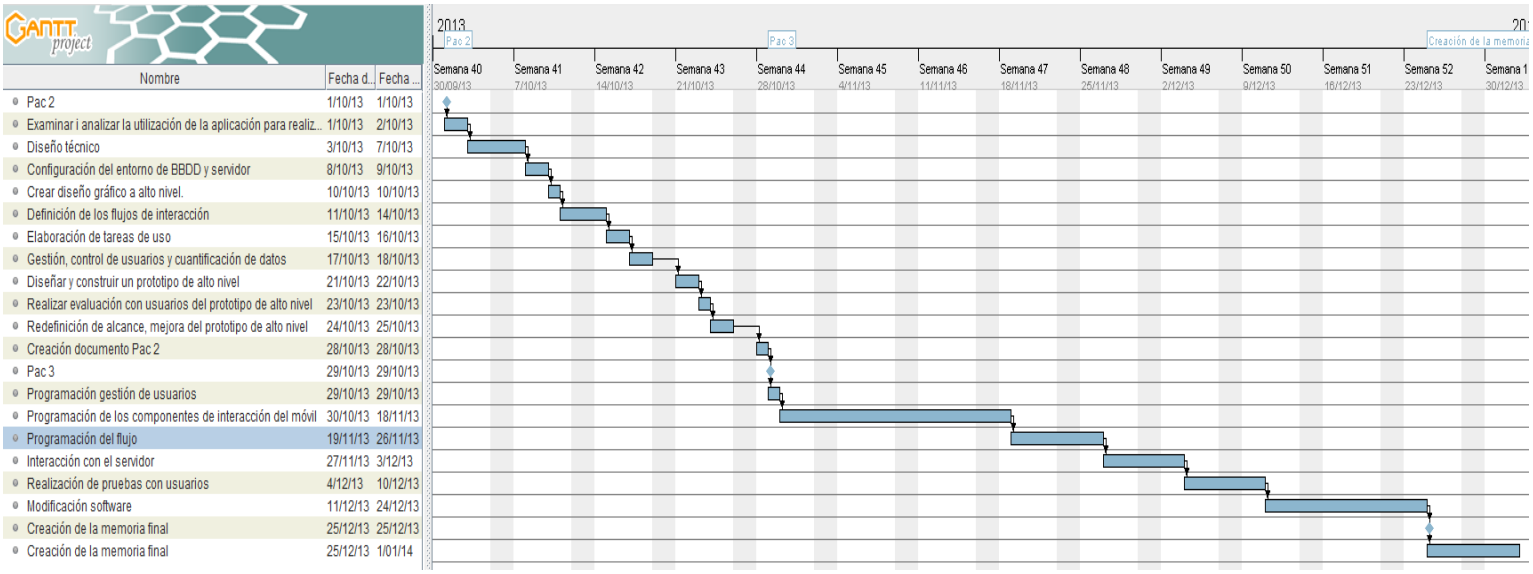
Planificación

Sobre el calendario ubicado en la asignatura se realiza una planificación que tiene en consideración las siguientes fases:

- Definición del trabajo y planificación.
- Definición de los requisitos.
- Análisis y diseño de la aplicación.

- Implementación.
- Test Unitarios y Pruebas.
- Preparación de la documentación final.

Para tener una esquematización gráfica y más visual que la típica tabla de planificación se realiza un diagrama de Gantt.



Nombre	Fecha d...	Fecha ...
Pac 2	1/10/13	1/10/13
Examinar i analizar la utilización de la aplicación para realiz...	1/10/13	2/10/13
Diseño técnico	3/10/13	7/10/13
Configuración del entorno de BBDD y servidor	8/10/13	9/10/13
Crear diseño gráfico a alto nivel.	10/10/13	10/10/13
Definición de los flujos de interacción	11/10/13	14/10/13
Elaboración de tareas de uso	15/10/13	16/10/13
Gestión, control de usuarios y cuantificación de datos	17/10/13	18/10/13
Diseñar y construir un prototipo de alto nivel	21/10/13	22/10/13
Realizar evaluación con usuarios del prototipo de alto nivel	23/10/13	23/10/13
Redefinición de alcance, mejora del prototipo de alto nivel	24/10/13	25/10/13
Creación documento Pac 2	28/10/13	28/10/13
Pac 3	29/10/13	29/10/13
Programación gestión de usuarios	29/10/13	29/10/13
Programación de los componentes de interacción del móvil	30/10/13	18/11/13
Programación del flujo	19/11/13	26/11/13
Interacción con el servidor	27/11/13	3/12/13
Realización de pruebas con usuarios	4/12/13	10/12/13
Modificación software	11/12/13	24/12/13
Creación de la memoria final	25/12/13	25/12/13
Creación de la memoria final	25/12/13	1/01/14

Análisis y diseño de la aplicación

Elección de los métodos de indagación

La aplicación Regisbet es una aplicación para registrar las apuestas que se hacen entre los amigos.

Tendrá una interfaz sencilla, control de usuarios y consulta de apuestas.

Pero para ver realmente si lo que pienso desarrollar está cerca de lo que el usuario quiere primero se deben hacer unas indagaciones. Está investigación consta de:

- Comunicar al usuario el tipo de aplicación que se quiere realizar y preguntar al usuario como lo que quiere o como le gustaría que fuera.
- Otra fuente de información es mirando la competencia.
- Realizar una buena labor de investigación para poder recoger información de un producto cercano al que el usuario desea.

La investigación consistirá en:

- Indagación contextual
- Análisis competitivo de los productos que hay actualmente en el mercado

- Entrevistas a los futuros usuarios
- También se realizará una encuesta.

Queda descartado el logging para realizar una actividad basada en el monitoreo de la " actividad de los usuarios, para recoger y analizar los datos del registro de su actividad en un sistema o sitio web puede llegar a ser muy costoso y no habrá suficiente desarrollado del " aplicación para poder realizarlo.

Indagación Contextual

Para la indagación contextual se realizará la técnica de Shadowing (MÉTODO DE SEGUIMIENTO) es un método de investigación cualitativa en la que el " investigador observa un participante (o más de " uno) mientras lleva a cabo sus actividades cotidianas.

Esta técnica es posible porque trabajo en una consultoría, ante el " ordenador con muchos más compañeros que utilizan el " ordenador de forma continua durante todo el día, de la misma forma también utilizan el móvil por motivos personales y algunos por motivos profesionales, por tanto un par de personas formarán parte de " este estudio durante un período de una semana.

También los preguntaré sobre sus hábitos por las noches y los fines de semana ya que en esa franja no puedo estar delante para poderlo "controlar".

Shadowing y Entrevistas de actividades cotidianas.

Hay muchos tipos de usuarios que utilizan el móvil de forma continuada durante el día.

Pero se han diferenciado 3 tipos dentro del contexto de investigación y entrevistas realizadas.

Más adelante se hará una descripción más detallada de cada uno de los tipos y usuarios que se han encontrado.

De momento los números de forma genérica como “El trabajador”, “El estudiante”, “El jubilado o casi”.

Como los perfiles de usuario de “El trabajador” se realiza un detalle de acciones.

Por la mañana no hacen uso del móvil excepto para levantarse, ver si va a llover y la temperatura en el exterior.

Principalmente cuando están en el trabajo a los usuarios por motivos de compromiso con la empresa utilizan el Whatsapp como herramienta de conectividad con el exterior. Durante periodos cortos y aproximadamente de unas diez a quince veces al día.

Al mediodía a la hora de comer se utiliza aplicaciones diversas de entretenimiento como top eleven, travian donde juegan durante un periodo de cinco minutos a 10 minutos.

Por la tarde cuando salen en el transcurso de vuelta del trabajo utilizan Whatsapp para conectarse con el exterior y utilizan juegos de entretenimiento como triviados, Candy crush saga y similares.

Por la noche aprovechando la conexión Wifi de casa y que tienen tablets y ordenadores se conectan para ver series, navegar por internet o consultar el correo personal y " empresa y sobre todo conectarse a páginas sociales como Facebook, Twitter.

El otro perfil de usuario llamado “El estudiante” no puede utilizar ningún app durante la mañana hasta que termina su jornada estudiantil. Desde entonces utiliza la conexión Wifi de casa y el móvil durante todo el medio día para conectarse a internet para ver páginas, jugar a alguna aplicación de " entretenimiento estilo angry bird, también comprueba continuamente el Whatsapp. Por la noche antes de ir a dormir utiliza el Whatsapp como medio de comunicación y finalmente antes de " ir a dormir realiza una pequeña partida en alguna app de juegos.

El último perfil de usuario “El jubilado o casi”, no utiliza el móvil por la mañana, trabaja o pasea o escucha la radio y únicamente lo tiene para recibir llamadas.

Por la tarde realiza varias llamadas y utiliza el poco el Whatsapp con los amigos. Es los fines de semana cuando utiliza más el móvil, realiza llamadas frecuentes aprovechando la gratuidad de éstas, realiza bastantes fotos de las actividades que realiza y lo comparte con los compañeros mediante Whatsapp.

Conclusión

Después de haber recogido la información se puede decir que sobre la muestra elegida el gran volumen de utilización es por la noche los días de cada día, con móviles, tablets u ordenadores convencionales.

Pero también hay durante todo el día una utilización frecuente de las aplicaciones de poco tiempo de utilización.

El fin de semana varía bastante pero sobre todo hay una gran utilización los domingos por la noche y sábados a media tarde, esta última por utilización de mensajería instantánea o llamadas.

Descripción de los perfiles

En este apartado se mostrarán los datos de las personas que van a valorar las aplicaciones de la competencia así como también darán opinión de como les gustaría que fuera la aplicación planteada.

Piedad Ribes

El usuario con perfil “Estudiante” se llama Piedad Ribes es de Mataró y esta realizando la carrera de Ingeniera Industrial en Terrassa y tiene 27 años.

Su frase es: ¡En este momento estoy en Los mejores Momentos de mi vida y cada día que pase continuara siendo el mejor momento!

Bibliografía: Estudiando los días de cada día, camarera de fin de semana y apasionada de los perro. Desde hace mucho que tiene un smartphone pero su conexión a internet siempre ha sido bastante limitada. La conexión a ADSL de casa se le hace vital para su día a día. Busca sobre todo información de la universidad porque así es más rápido, no espera que el computador se inicie.

Por ella hoy en día sería impensable vivir hoy en día sin su móvil.

Modelo móvil: Tiene un Samsung Galaxy Ace 2 con un Android 2.2.3. Usuaria medio-adelantada en los móviles.

Objetivos: Quiero grabar todas las cosas que mi pareja me dice. Después de 4 días siempre se le olvida que lo he dicho y me dice que le he puesto yo en su boca. Así que esta aplicación sea perfecta para demostrar que se comprometió a hacer una tarea que después olvida.

Comportamientos: Perfeccionista, inconformista social por naturaleza, amiga de los animales, amante del paddle sub y con continua renovación de noticias de actualidad.

Necesidades: Quiere tener una aplicación sencilla y que sea rápida de grabar la apuesta, pone especial mención en la grabación de la apuesta verbal más que escribirla. Indispensable poder realizar apuestas públicas y privadas.

Jorge Cancho

Representa el perfil de "El trabajador" se llama Jorge es de Barcelona y cercanías ha realizado la carrera de Ingeniería Técnica de Informática, tiene 33 años.

Su frase es: ¡Se Tiene que vivir la vida como el destino te la trae, depende de ti que quieras Verla como un sueño o dormido!

Bibliografía: Estudiante de FP de Grado superior, licenciado mediante la UOC y estudiando inglés en la actualidad.

Su cargo empresarial es autónomo (los pocos que quedan) ocupando el puesto de Jefe de proyecto de una empresa pequeña, trabaja 25 horas al día. Esta tantas horas en el trabajo que su casa es la segunda residencia. En un breve periodo de tiempo quiere que cambie y en breve dará más prioridad al ocio.

Quiere vivir a al máximo los momentos que le plantea la vida y conseguir el máximo de la vida.

Le gusta hacer deporte en la calle e ir al gimnasio, amigo de sus amigos una gran persona y además te saluda cuando te ve por la calle (como los homicidas y asesinos).

Modelo móvil: Tiene un Samsung Galaxy III con un S.O. Android 4.2.3. Conocimientos avanzados en la utilización del móvil.

Objetivos: Esta aplicación quiere que le brinde la oportunidad de hacer apuestas con los amigos para las maratones que quiere realizar y claro ganar. Quiere ganar cenas gratis ya que nunca quiere cocinar y la compañía de sus amigos le llena de gozo así que matará dos pájaros de un tiro.

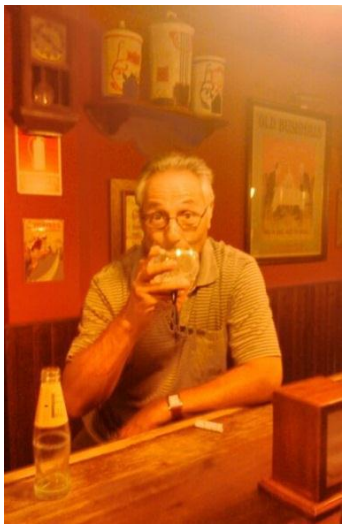
Comportamientos: Conformista, con ganas de vivir experiencias nuevas, deportista amateur, futbolista aficionado, amigo de los chistes.

Necesidades: Sobre todo que sea fácil y rápido realizar la apuesta. También aportaría un valor añadido el tener una parte de chat, ya que así se podrá interactuar de forma mucho mejor.

Jose Luis Carrasco

Persona de 60 años padre de familia y abuelo, esta aprendiendo las nuevas tecnologías con la tan conocida técnica de prueba -error, aunque haya más error que acierto.

Tiene como filosofía gastarse mucho dinero en los gadgets que va adquiriendo y se piensa que contra más dinero cuesta el aparato, mejor le irá.



Tiene como filosofía de vida, tomárselo todo con calma y si puede ser con alguna cervecita que otra mejor será la velada.

Quiere hacer del mundo un lugar mejor pero le molesta que se tenga que mover, así que hace todo lo que puede pero como el anuncio de los cubanos, sin estresarse.

Su pasión es el fútbol, el fin de semana que no hay fútbol, no hay fin de semana.

Pasa todo el tiempo que puede con su nuevo juguete, el principito de la casa. Su nombre se Luis y es su limpio, todo a pesar de todo gira en torno a él, tanto que incluso los padres están cansado de los abuelos para pesados .

La frase que mejor le representa es: ¡Yo lo que quiero es que estéis todos sanotes y sobretodo seáis felices!

Bibliografía: Trabajador desde hace mas de 45 años, siempre trabajando de lo que saliera y al mejor postor. Desde hace mas de 20 años está trabajando en una empresa de perfumes, su cargo es el de ayudante químico.

Modelo móvil: Tiene un Sony Xperia Z con un S.O. Android 4.2.1. Tiene conocimientos muy básicos en la utilización del móvil.

Objetivos: No tiene ningún objetivo con la aplicación, no la utilizará.

Comportamientos: Tranquilo, sin muchas inquietudes, con ganas de hacer reír a todo el que esté cerca de él pero siempre comportando la apariencia.

Necesidades: No la utilizara pero si la utilizas le gustaría que fuera lo más sencilla posible, o que se parezca a alguna que ya tiene, así le será mas fácil acostumbrarse a ella.

Análisis Competitivo (BENCHMARKING)

Para resolver este apartado se debe comprobar cuáles son las aplicaciones que hay en el mercado tanto en el mundo de app como en la Web.

He realizado una búsqueda mediante el market de Android y se ha encontrado aplicaciones para hacer apuestas mediante dinero o simplemente pronósticos deportivos , con aplicaciones como "bet friend "," bet365 "," beticious ", esta ultima cerró el 30/11/2013 .

Estas necesitan de un registro de usuario.

Al realizar una búsqueda por aplicaciones donde hay apuestas con amigos las dos que he encontrado necesitan de conexión vía Facebook. Una se Android app y " otro es una web con conexión vía Facebook.

El nombre de la aplicación Android es " Let's Bet" y el nombre de la aplicación web es Betsy.

Beticious

Actualmente esta aplicación ha cerrado.

Beticious es una aplicación para hacer apuestas deportivas, de deportes de todo el mundo y diferentes niveles y opciones profesionales.

Lo que más ha gustado de esta aplicación Android es que puedes registrarte mediante Facebook o por registro propio, por lo tanto puedes elegir cómo hacer la " entrada al " aplicación.

Puedes recibir regalos a medida que vas acertando pronósticos deportivos y ganas puntos.

Para probar la aplicación se creará una cuenta nueva.

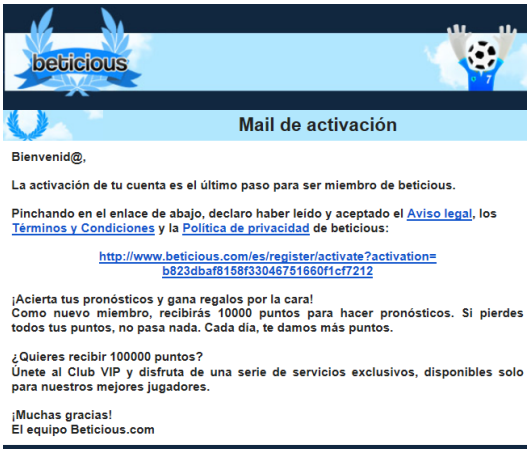
Para ello se pone el nick, la e- mail, la contraseña y se confirma el link que te "envían.

Si se hace el registro mediante el registro propio tienes una nueva cuenta, pero ningún registro. En cambio si haces el registro mediante Facebook puedes " invitar " a todos tus contactos de forma mucho más rápida y fácil.

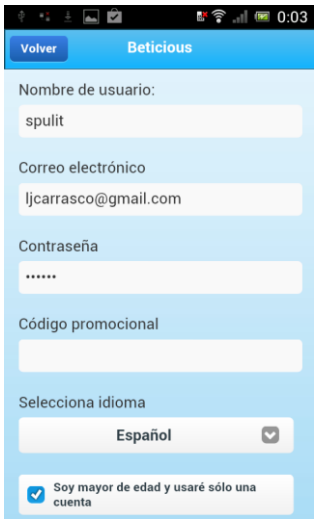
Creo que es una opción un poco más atractiva porque podrás invitar a tus amigos a hacer apuestas de forma mucho más rápida.

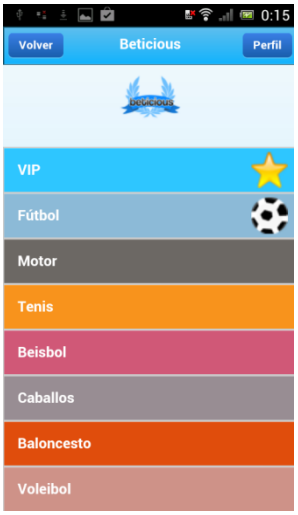
El login es bastante sencillo.

Introduces nombre, password y e-mail. Te dan un link de confirmación y ya puedes empezar a realizar apuestas.



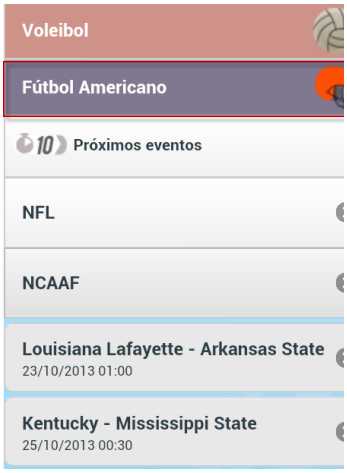
Al realizar la entrada en la cuenta puedes ver todas las opciones para apostar que te ofrece la aplicación.

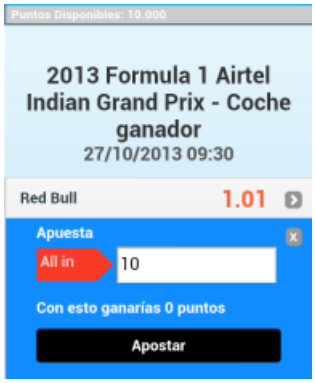




Se pueden hacer apuestas de todos los deportes que son mínimamente conocidos, aunque la elección de colores no es la más idónea, se ve una clara diferenciación clara de todos los deportes.

Para realizar las apuestas solamente hay que hacer clic sobre el deporte que quieres hacer las apuestas. Te muestra las más cercanas en el tiempo y de qué liga quieres apostar.





Finalmente te muestra que la apuesta se ha guardado correctamente, con lo que puedes continuando haciendo apuestas hasta que se te agota los points.

Los Points que se pudo ganar si:

- Se compran con dinero

- Se ganan las apuestas realizadas.

- Se realizan descargas de app de Android o se visualizan vídeos promocionales.

- Se compran ciertos productos que además de recibirlos ganas points.

Conclusión Beticious

Creo que es una aplicación muy completa, sencilla y aditiva.

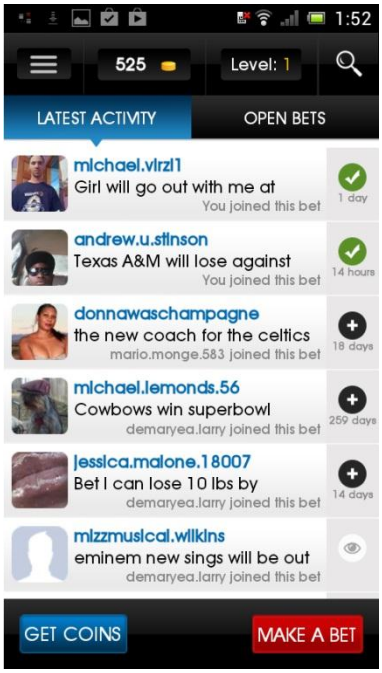
Que se puedan ganar regalos o los puntos los puedas cambiar por objetos hace adicto la app.

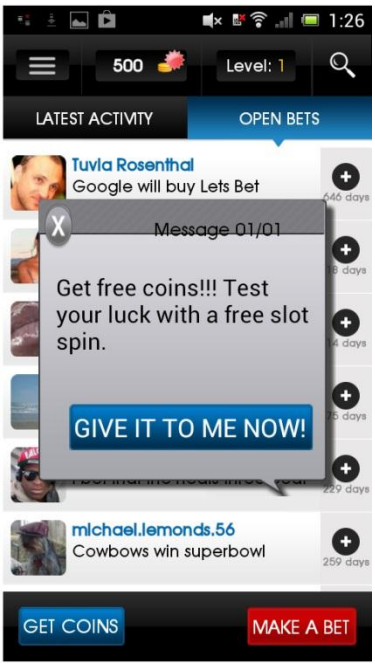
La interfaz es sencilla e intuitiva, aunque los colores de presentación inicial no atraen la atención del jugador al resto de menús hace que sea idóneo para ver las diferentes opciones que hay.

Let 's Bet

Aplicación de Android para hacer apuestas entre amigos, muy parecida a la que se quiere desarrollar.

Se baja la aplicación que te pide la cuenta de Facebook para poder ingresar a la aplicación, con la que cogerá tus contactos y permitirá poder realizar invitaciones para hacer apuestas.





Después de insertar los datos de acceso el que se muestra es un pop-up de cómo ganar "coins".

Se acepta el pop-up y se muestra un listado de apuestas abiertas en que todos pueden hacer apuestas.

Para hacer una apuesta se puede hacer clic sobre el botón rojo de la derecha "make a bet" donde muestra una nueva pantalla.

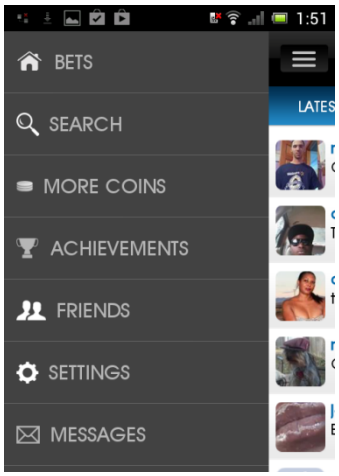
Se indica la apuesta, la cantidad de coins, a quién va dirigido, donde puedes escoger de la agenda personal o de Facebook y las opciones que hay.

La introducción no es muy intuitiva y no es una experiencia muy amigable. Además el pop-up del principio hay veces que no responde y siempre se queda en pantalla sin poder hacer nada al respecto.

Lo que más ha gustado es que hay apuestas abiertas en que todo el mundo puede opinar y votar.

Hay dinero virtual que puedes ganar haciendo ciertas "Misiones" como alquilar tiene un número de veces, acertar un número determinado de apuestas, etc.

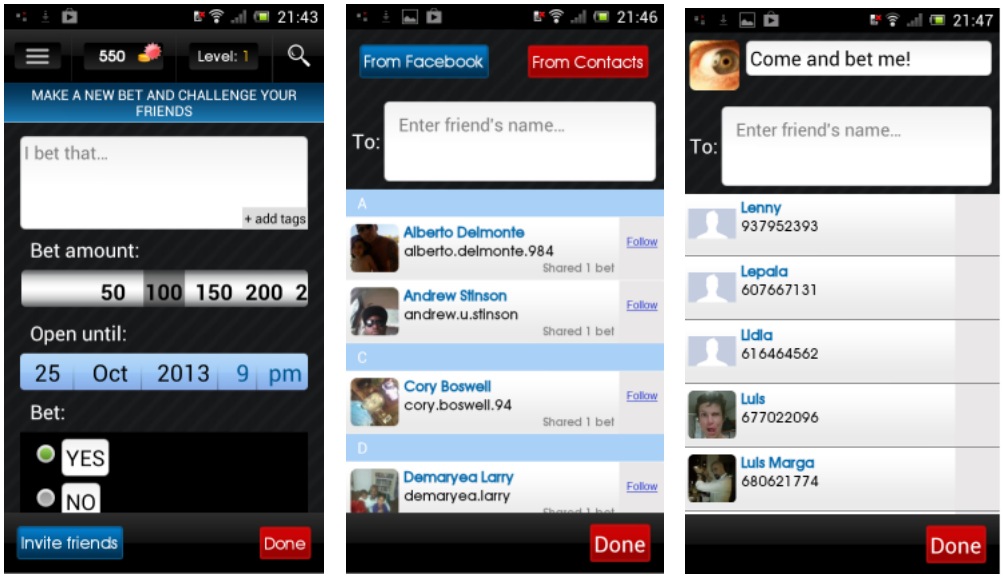
El menú es sencillo e intuitivo.



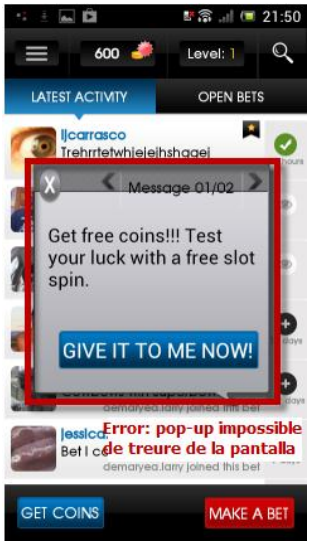
Accedes rápidamente a las opciones de la aplicación.

Son pocas pero no necesita más.

A la hora de realizar la apuesta es sencilla.



Te marca la apuesta que se ha de realizar, los tags para facilitar búsquedas, creo que es una opción totalmente innecesaria, las personas que están implicadas en la apuesta, la fecha de finalización, cuando se termina y las respuestas posibles.



Cuando pulsas sobre el botón dé, realiza el alta de la apuesta y te va directamente a la lista de apuestas, pero aparece un pop-up que me ha sido imposible sacar. Es un bug y además bastante grande.

Para continuar jugando, en caso de que te quedes sin coins se puede comprar pagando o haciendo pruebas como logar tantas veces, realizar / acertar varias apuestas...

Conclusión Let's Bet

Aplicación muy parecida a la que se quiere desarrollar, tiene la misma esencia que se quiere tener para la aplicación a desarrollar.

Se clara y se pueden crear apuestas de forma fácil y sencilla.

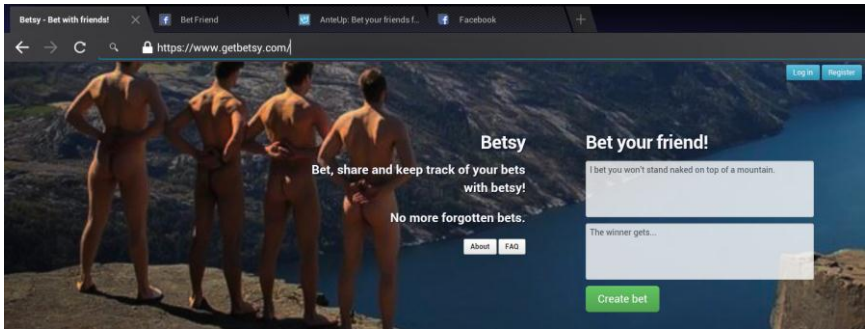
El diseño falla un poco, sobre todo cuando la aplicación muestra pop-ups con información hay veces que por mucho que pulses la pantalla no responde.

Gusta que se puedan ver las fotos que tiene la gente y que se pueda invitar desde la agenda.

No me gusta nada que se necesite la cuenta de Facebook para poder entrar a la aplicación

GetBetsy

Tiene una presentación como mínimo original que hace que durante treinta segundos prestes atención a lo que hay en la pantalla.

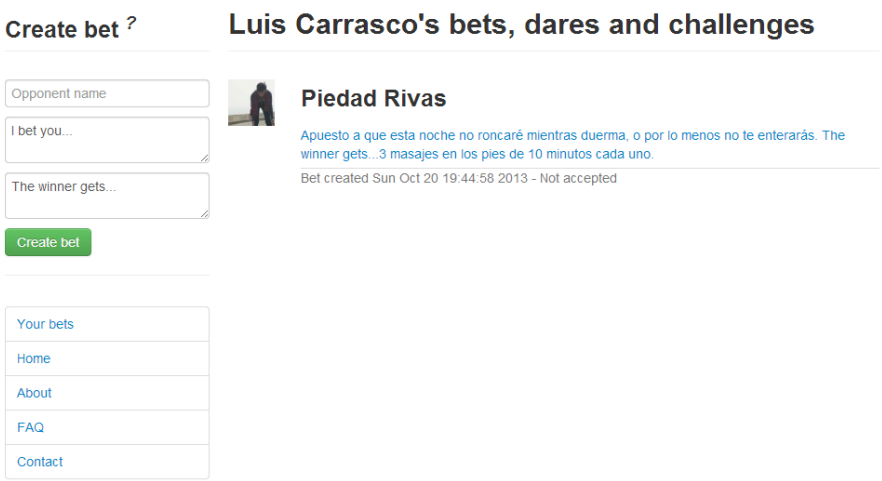


A parte de la imagen graciosa hay dos campos de texto y un botón, no hay duda de que hay que hacer y por lo tanto por esta parte es bastante intuitiva.

Se llena los dos campos y se pulsa sobre el botón "Create bet" a partir de aquí la web pierde su encanto.

Necesita conexión con Facebook, lo que a muchas personas les tira hacia atrás, como por ejemplo el usuario 4 que cuando lo vio no quiso continuar. Aun así seguimos y se ve que la interfaz sigue siendo sencilla.

Mantiene los dos campos de texto, un campo nuevo donde indica a quien va destinada la apuesta y el botón de crear la apuesta.



Contiene una interfaz muy clara y sencilla, ideal para móviles, ya que te conectas mediante Facebook donde también es una aplicación que se puede ver mediante navegador.

Conclusión GetBetsy

Aplicación Web sencilla, tanto que se para hacer apuestas con amigos por Facebook, no existe app pero se puede conectar mediante navegador a esta web.

La más sencilla de las que hemos encontrado y con un diseño muy acertado, se para tenerlo en cuenta.

Por el contrario se echa en falta campos más gráficos, insertar alguna imagen o audio, también falta la gestión de contrincantes porque solo puedes crear una apuesta contra otra persona.

Le falla la necesidad de hacer login con Facebook para poder funcionar.

Análisis Competitivo con usuarios

Se realizarán las pruebas de las aplicaciones con los usuarios para comprobar que se lo piensan y así aportar otro punto de vista que se pueda reaprovechar.

Beticious por Jose Luis Carrasco

La primera frase ha sido:

"No se ‘paque’ me haces probar esto si no lo voy a utilizar”.

Aun así se esfuerza porque le digo que se para el proyecto de final de carrera.

Cuando ve que la app se de deportes ya le gusta más, es un tema que domina y motiva.

Se hace un lio con el registro, el Facebook sabe lo que se embargo, no tiene ni quiere tener un perfil.

Gusta: Los colores son llamativos y quedan claros. Es una aplicación sencilla y amigable, aunque él no se sale demasiada ve que no es difícil de usar.

No gusta: No le gusta los links que envían a la e- mails para poder verificar la cuenta, se pierde tiempo.

Destacable: el poder ganar regalos a pesar de ser muy complicado. El poder hacer apuestas muy grandes y muy pequeñas. Que hay una gran variedad de apuestas posibles.

Let 's Bet por Jorge Cancho

Lo más significativo desde el inicio de la prueba ha sido:

"No me gusta nada el hecho de que me haya de logar mediante Facebook”

Los pop -ups le molestan y no quiere saber nada de las coins que te dan.

Le gusta que se pueda " chatear" y el diseño lo encuentra simple. No se le ven bien los datos para poder poner fin a la apuesta así que finalmente y sin más remedio puso una fecha de 15 de mayo de 2020...

Gusta: Menú sencillo, diseño intuitivo, la máquina tragaperras para gana coins no se la esperaba pero no le disgusta.

No gusta: Los coins cree que sobran , los pop -ups los encuentra muy intrusivos , no ve clara como introducir el texto de la apuesta y sobre todo no se bajaría nunca esta app por el hecho de necesitar registro para Facebook es algo que le molesta muchísimo.

Destacable: las apuestas son fáciles de hacer, el menú principal tiene opciones estándares.

GetBetsy por Piedad Rivas

La primera frase es: "¡Ummmmm la foto está bien!”.

Sólo entrar en la aplicación ha aceptado la apuesta y su juicio ha cambiado.

Ha intentado subir una foto, arriba de todo está el botón, no le gusta como la sube es demasiado grande y eso que la foto era pequeña...

Además indica que los botones están en la parte de arriba, desde cuando están en la parte de arriba, nunca se queja.

De la apuesta, se declara perdedora, pero sorprendentemente le pone que ha ganado, se indigna dice que no la entiendo que es muy complicada y abandona la aplicación, se alegra porque a ganado pero no era lo que quería.

Gusta: La foto inicial le ha llamado la atención.

No gusta: No le gusta nada, no la entiende. Se pone como perdedora y después sale como ganadora....

Destacable: Diseño sencillo pero se la única cosa que tiene buena.

Encuestas

Hacer encuestas permitirá principalmente ver si se utilizará la aplicación y las tareas más importantes que la gente quiere que tenga la app.

También sirve para ver cómo, cuándo y dónde se utiliza el móvil las personas y en consecuencia la app.

Sobre todo se puede ver el impacto que tendrá tu aplicación en el día a día de esta persona, si lo utilizará y que espera de ella.

El formulario se ha realizado en castellano para llegar a la máxima población posible.

El método de promoción para que se haya realizado la encuesta ha sido por Facebook y Whatsapp.

Por falta de recursos y tiempo se ha realizado una no muy complicada encuesta donde se indican las preguntas con las respuestas dadas por los encuestados.

Formulario encuesta

La encuesta se ha realizado mediante Google docs., ya que es un sistema muy sencillo de creación de formulario con el que se puede tener análisis de las respuestas de forma inmediata.

El formulario se puede encontrar en:

https://docs.google.com/forms/d/1eyFQIrr0fOepUl7f5h8cAI0UYKMDIKqIbwzIl9GRntM/viewform

Edad

Con esta pregunta se intenta saber en qué franja de edad del usuario.

Edad

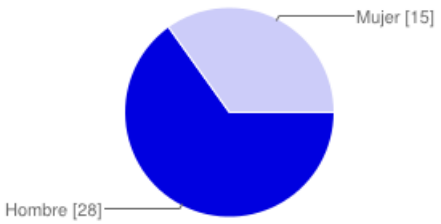
35 36 33 34 39 38 41 40 23 24 27 28 29 30 32 31

Hay más registros de usuario dentro del mercado de Apps pero en mi caso mis contactos tienen la franja que va desde los 23 hasta los 41.

Sexo

Ver cuál es el porcentaje de personas que utilizan el smartphone, aunque esta parte no es real, en mi entorno hay más hombres que mujeres.

Sexo

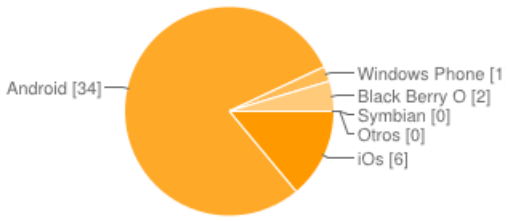


Hombre	28	65%
Mujer	15	35%

Sistema Operativo

Se quiere saber cuál es el SO más utilizado dentro de las personas cercanas que realizan la encuesta.

Sistema Operativo del Smartphone



iOs	6	14%
Android	34	79%
Windows Phone	1	2%
Black Berry OS	2	5%
Symbian	0	0%
Otros	0	0%

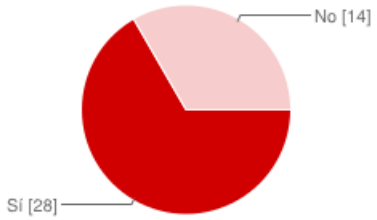
Hay mayoría de personas que tienen Android, esto es bueno porque es el SO con el que se creará la aplicación, como se ha visto anteriormente si tuviera un entorno más exclusivo puede ser que la franja de iOs sube pero no es el caso.

Utilizando la aplicación para registrar los retos con los amigos

Es quiere saber si hay mercado interesado dentro de la población.

Creo que la encuesta como se ha hecho propaganda con personas cercanas y éstas siempre quieren lo mejor para ti, el pronóstico que sale es más optimista de lo que realmente será en la realidad.

Utilizarías la aplicación para registrar tus apuestas con los amigos?



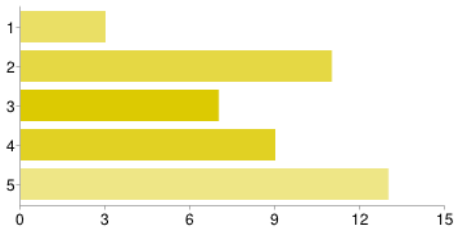
Sí	28	67%
No	14	33%

De las 42 personas encuestadas tal que el 67% utilizas la aplicación al ser contactos cercanos y al saber que es una app para el proyecto de final de carrera creo que han mentido y esta proporción quedaría en la mitad un 33% que implica 14 personas de las 42 iniciales.

Acciones más importantes en la aplicación

Se quiere saber de las acciones básicas que tiene la aplicación cuál es la más importante para los futuros usuarios.

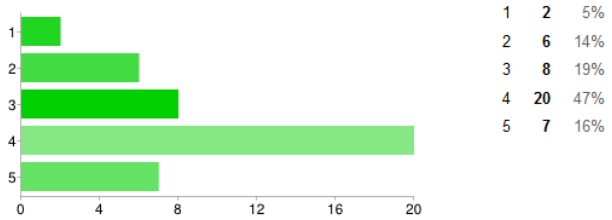
Registro de contactos [A que le das más importancia en la aplicación?]



1	3	7%
2	11	26%
3	7	16%
4	9	21%
5	13	30%

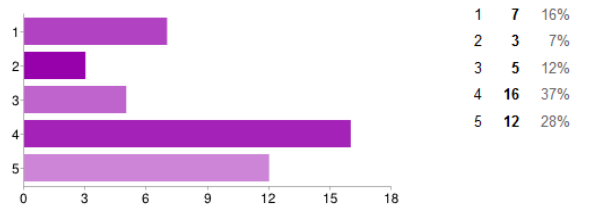
Hablando con los futuros usuarios me he dado cuenta que no hace falta tener un registro de usuario. Con un control de usuarios como hace Whatsapp ya es suficiente y mucho más rápido.

Registro de apuesta de forma rápida sin contactos [A que le das más importancia en la aplicación?]



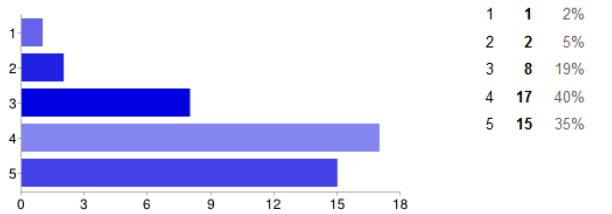
Como se ha comentado en el apartado anterior las personas deciden que no es importante tener un control de usuarios.

Insertar fotos, vídeos o audio para registrar la apuesta [A que le das más importancia en la aplicación?]



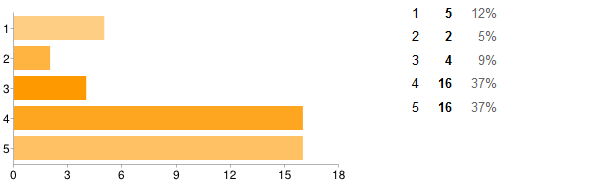
Para mí es una parte bastante importante, hace que la aplicación pueda registrar de forma más rápida el pacto contractual entre las dos partes.

Ver todas las apuestas de tus amigos [A que le das más importancia en la aplicación?]



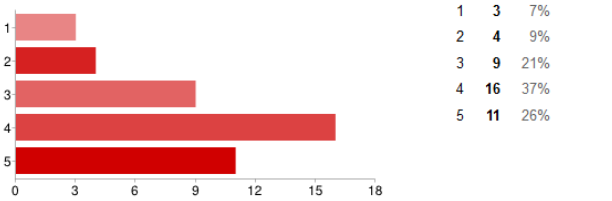
El poder cotillear todas las apuestas de los amigos, así como poder comentar para así subir el ambiente de la apuesta es un tema bastante interesante para los usuarios.

Tener apuestas privadas/ocultas [A que le das más importancia en la aplicación?]

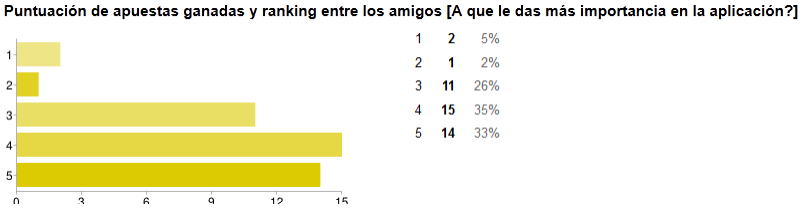


De igual forma el hecho de tener las apuestas con tu colectivo de amigos sin que otros se den cuenta, también es un tema importante y por tanto a tener en cuenta.

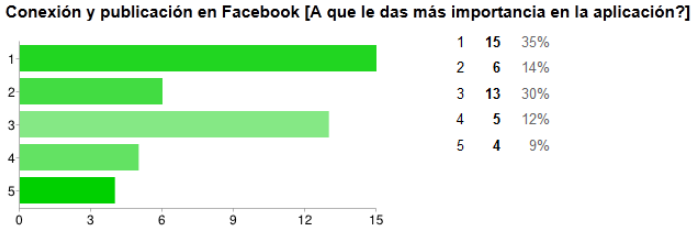
Configurar para recibir avisos recordatorios [A que le das más importancia en la aplicación?]



Saber que falta poco para terminar de cumplir la apuesta o que vaya indicando de forma clara cuando se debe tener finalizada la prueba es un tema importante.



Hablando de una aplicación en que apuestas cosas, también es normal que la gente pueda demostrar cuáles son las apuestas que ha acertado y cuáles no. Mostrando un ranking de todos tus amigos.



Para usuarios no es importante que la gente vea las apuestas realizadas. Por lo tanto creo que es un apartado que se caerá en la realización de la aplicación.

Creo que falta...

Campo de texto libre para que la gente pueda opinar que es lo que hace falta y le gustaría que estuviera.

El comentario más destacable aunque igualmente ignorable es el de chicas desnudas, quizás el sector masculino le guste, pero perdería al género femenino y no tendría nada que aportar dentro de la aplicación.

Los comentarios que se pueden tener en cuenta es la pasarela de pago y Bitcoin para apostar.

Otros comentarios que me gustaría realizar aunque son imposibles de llevar a cabo es el de hacer que los amigos contesten rápido sin tener que esperar y tener la posibilidad de apostar con amigos de amigos.

Conclusión

La encuesta me ha ayudado a ver más claro que es lo que la gente quiere y no quiere, porque además de la encuesta la gente preguntaba por el canal de emisión (Facebook y Whatsapp) y también aportaba más ideas que no una fría encuesta.

Así pues he llegado a las conclusiones que:

- No habrá pasarela de Facebook para gestión de usuarios.

- Los usuarios se tratarán estilo Whatsapp, por control de agenda.

- No habrá coins, ni points, ni ningún tipo de moneda virtual.

- Se realizará apuestas privadas y públicas.

- Habrá un chat para " amenizar " la apuesta.

Información complementaria a la encuesta

Aun así para ampliar información se ha buscado por internet de encuestas y estadísticas más externas ya realizadas donde la conclusión es mucho más amplias en el formulario realizado y semejante a las lecturas recomendadas establecidas:

- Los españoles utilizan sus smartphones preferentemente y en orden de mayor a menor uso desde: casa, allí donde van, cafeterías, trabajo, restaurantes y transporte público

- Las actividades que realizan los españoles al mismo tiempo que utilizan sus smartphones son principalmente escuchar música o la radio, o ver la televisión

- El usuario medio tiene 19 aplicaciones instaladas, 6 aplicaciones pagadas y 8 utilizadas en el último mes

- El 79% de ven videos con sus smartphones y el 23% lo hacen todos los días

- El 79% de usan redes sociales a través de sus smartphones y el 51% lo hace al menos una vez al día

- El 24 % de los usuarios de smartphones españoles utilizan sus dispositivos para hacer compras

- El 43% realiza compras a través de sus ordenadores, el 27% en la tienda y el 22 % desde su Smartphone.

La utilización de los Smartphones más gráfica es:

	Actividad	min/día	% tiempo
APPS	Redes Sociales	17	63%
	Escuchar música	16	
	Juegos	14	
	Emails	11	
	Ver TV, películas	9	
	Leer libros	9	
	Hacer fotos	3	
WEB	Navegar	25	20%
STD	Hacer llamadas	12	18%
	Mensajes SMS	10	

Datos de estudio de Telefónica Europa (Jun 2012) presentado por Javier Lorente en el evento Mobile Day:

Se indica que el mercado de Android, tanto en compra de dispositivos como market está aumentando de forma considerada trimestre tras trimestre.

Un estudio realizado por Pew Internet & American Life Project reveló que en mayo de 2013 un 28 % de los usuarios utilizan el sistema Android, frente a un 25 % que posee un iPhone.

El estudio también analiza las diferencias entre los usuarios de estos dos SO dependiendo de su edad, sexo o situación económica.

Los resultados son curiosos:

- Las mujeres utilizan por igual iPhone y Android con un 26%

- Los hombres prefieren Android por siete puntos con un 31 %.

A medida que va aumentando la edad la preferencia por Android va disminuyendo su porcentaje:

- Los usuarios de entre 18 y 24 años utilizan Android en un 43 % frente a un 31% en iPhone.

- A medida que avanza la edad disminuye la diferencia y en los usuarios de más de 65 años iPhone es el ganador con un 11 % frente a un 7% de Android.

Un dato que me ha llamado la atención es que se demuestra que los más ricos prefieren iPhone.

Perfiles identificados

La aplicación tiene diferentes usuarios, en una apuesta hay diferentes roles a adoptar:

- Los administradores de la apuesta que son también los creadores.

- Los que son retados.

Todos tienen su funcionalidad y perfiles dentro de la apuesta.

Características del perfil.

Administrador/es o Creador

Perfil / es de usuario que se encarga de realizar y configurar la apuesta.

Tiene permisos totales sobre la apuesta y puede realizar cualquier acción.

Invitará a todas las personas de la agenda que haya estilo Whatsapp, pondrá una foto de perfil para la nueva apuesta (opcional), creará un nuevo texto redactado de la apuesta en cuestión, podrá subir cualquier documento gráfico como por ejemplo fotos, también establecerá fechas de finalización.

Finalmente podrá realizar una votación de cuál es la persona / s que han resultado ganadoras.

Además podrán realizar todas las acciones del resto de roles que a continuación se mencionarán.

Invitados o retados

Podrá ver cuál es el estado de la apuesta.

No podrá realizar ninguna acción sobre la configuración de la apuesta.

Podrá observar cuál es el avance de la apuesta.

Son los perfiles que deben superar el reto propuesto. Tendrá el mismo rol que el observador.

Características demográficas

Realmente no hay ningún tipo de limitación o apartado que deba añadir especialmente para tener en cuenta en la demografía de los usuarios.

Se independiente de donde sean ya que cualquiera de ellos podrá aportar su granito de arena dentro de la apuesta aunque no estén demográficamente juntos.

Está claro que si una prueba se debe realizar físicamente debe tener un lugar donde se ha de realizar y por tanto debe haber el retado como el retador como mínimo.

Pero por ejemplo si la apuesta no necesita de prueba material como puede nada por ejemplo aprobar el proyecto de final de carrera, basta demostrar que se aprueba con la entrega.

Por lo tanto no tiene ninguna limitación geográfica, únicamente debe traducirse en diferentes idiomas para que se pueda utilizar en cualquier parte del mundo.

Intereses y Motivaciones

La motivación o interés en utilizar esta app de ocio la tendrá aquella persona se le plantea un reto y quiere que quede constancia de ella. Por lo tanto se pudo demostrar mediante un medio contractual que se puede realizar o no una apuesta.

Así los términos de la apuesta no quedan dispersos, a la finalización de la prueba se puede demostrar con más certeza que es lo que ha pasado y los motivos de la decisión final de la apuesta.

Debe haber al menos dos personas implicadas para que la aplicación tenga una utilidad.

Experiencia con el uso de la tecnología móvil

La aplicación está pensada para un tipo de usuario con conocimientos de Android medio-bajo.

Será intuitiva y no tendrá gran complejidad en cuanto a tareas y escenarios de uso.

La gestión será muy parecida a la de Whatsapp que es una aplicación que en el mercado español tiene una gran tasa de mercado, pero estará orientado a las apuesta

Examinar y analizar el contexto de uso.

Actualmente dentro de España el porcentaje de personas que no tienen un smartphone es bajo.

Casi la totalidad de los smartphones tienen conexión a internet y por lo tanto el market del sistema operativo.

El SO con el que está creado se Android por tanto el mercado de personas que pueden utilizar están limitadas a éste, que según las estadísticas es un número muy alto.

Actualmente los smartphones son pequeños ordenadores que tienen conexión las 24 horas al día y el usuario puede acceder siempre que quiera y donde quiera.

La utilización de Regisbet no es muy extensa en el tiempo, serán conexiones cortas y puede que haya mucha comunicación, se realizarán repetidas conexiones.

La utilización se producirá con los amigos, al producirse discusiones sin importancia, con los familiares, incluso con la pareja (puede ser un arma de doble filo).

Como hemos comentado antes, es una aplicación para que quede grabada la apuesta, para poder demostrar cuál fue el pacto, que se quería hacer y que pasó realmente.

Esta aplicación al ser de ocio, se utilizará en momentos puntuales y durante un corto período de tiempo, menos de cinco minutos son suficientes, acompañado con amigos o familiares en un entorno distienden.

Análisis de tareas

En este apartado mencionaremos qué conjunto de tareas necesitarán los usuarios para alcanzar sus objetivos en la aplicación.

Login
Este apartado será el más transparente para el usuario ya que no existirá.

Solo deberá bajar Regisbet para poder empezar a hacer apuestas con sus contactos.

Creación apuesta
Los campos mínimos para realizar la apuesta será:

- Redacción de la apuesta.

- Contra quién va dirigido, las personas podrán elegir los contactos de la agenda.

- Fecha de finalización.

- Insertar información gráfica adicional, este apartado es opcional.

Consulta de apuesta
Las apuestas serán públicas o privadas, en caso de que tengan privilegios de consulta podrán visualizar los datos de la apuesta pero no podrán realizar ningún cambio.

En las apuestas públicas todos pueden hacer de invitado. En las privadas se necesita previa invitación siendo uno con los que se apuesta en contra.

Un usuario también puede consultar y participar en el chat de las apuestas finalizadas en las que ha participado el usuario.

Finalización apuesta
Los creadores de las respectivas apuestas cuando llegue la fecha de finalización podrán introducir cuál ha sido el resultado. Cuando se conteste el resultado podrá variar hasta que se hayan completado las respuestas de los creadores.

Una vez pasada la fecha de la apuesta esta pasará a ser una apuesta finalizada.

Listado de apuestas
Existirán varios listados de apuestas un listado será el público y el otro listado serán las apuestas en que la persona ha participado y finalmente las finalizadas.

Apuesta Pública
- Todas las personas de la aplicación podrán acceder a las apuestas públicas de los amigos.

- Si se quiere también se puede acceder a todas las apuestas vigentes de la aplicación.

- Podrá ver cuál es el avance de la apuesta.

Apuesta Privada
- Solo se podrá acceder a la apuesta privada previa apuesta contra.

- Podrá ver el avance de la apuesta.

Apuesta finalizada
Una apuesta pasa a estar finalizada cuando la apuesta ha pasado la fecha y tiene como mínimo un resultado de un creador.

Cuando se cumplan estos requisitos la apuesta pasará a esta lista.

Configuración de Usuario

Es una acción no obligatoria que consta en modificar el perfil básico que se crea al principio de la utilización.

El perfil que se crea por primera vez es básico, pero estará vacío deberá ser rellenado por el usuario.

Este apartado se permite rellenar los datos como nombre, fotografía o e -mail.

Características descubiertas que deben estar presentes

En la fase de indagación se ha descubierto que poca gente quiere el login con Facebook y una creación de usuario, por lo tanto no se creará un login de usuario.

A partes iguales hay gente que quiere tener monedas virtuales como las que no.

Escenarios de uso

Para tratar este apartado se tendrá en cuenta ciertos criterios para plantear situaciones hipotéticas que se pueden dar en la vida real.

Estos usuarios pueden encontrar haciendo uso de la aplicación sobre condiciones especiales que provoca descubrir objetivos, funcionalidades y deseos de los usuarios en un contexto y situación determinada.

Por ello se tendrán en cuenta los siguientes aspectos a la hora de plantea escenarios de uso:

- ¿Qué perfil de usuario interviene?

- ¿En qué contexto se encuentra (dónde, cuándo , por qué ...) .

- ¿Cuál o cuáles son sus objetivos?

- Las tareas que lleva a cabo para la consecución de sus objetivos.

- Sus necesidades de información.

- Las funcionalidades que necesita.

- Como desarrolla estas tareas.

Escenario 1

Una familia algo diferente junta el fin de semana para ver el Barça- Madrid.

Después de unas cuantas cervezas y que el partido está a punto de comenzar en Florentino Madridista de toda la vida y orgulloso de serlo, pensando que este año si consigue ganar al Barça.

Reta al Sandro, este año si emocionado dice, a que el Madrid le mete 5 al Barça Sandro deberá invitar a él ya sus respectivas parejas a cenar, de lo contrario ya que se " posible " pero

improbable la cosa queda en unos cubatas en casa de Florentino y así buscan una excusa para volver a ver.

Perfil: Florentino será el administrador de la apuesta, Sandro el retado y las parejas las observadoras.

Objetivos: Se debe realizar una apuesta desde el principio.

Información: las parejas aprovechando que no les gusta el fútbol se van de tiendas y aprovechar que todo está vacío porque todo el mundo está en casa por el partido.

Tareas realizadas:

- Entrar en la aplicación.

- Pulsar el botón de nueva apuesta.

- Hacer la redacción de la apuesta, invitar al Sandro para que acepte la apuesta e invitar a las parejas.

- Pone como fecha límite el mismo día y la finalización de la apuesta será a las 20:45 ya que habrá terminado el partido y se sabrá el resultado.

Desarrollo: Sandro acepta la apuesta y pone un comentario en el chat, inserta una imagen con una mano y otra con la mano y un dedo subido.

Las parejas comentan el partido todo picante al florentino ya que es un resultado bastante improbable.

Florentino a medida que avanza el partido deja de hacer comentarios en el chat... cada vez es más difícil que gane la apuesta.

Cuando finaliza el partido en florentino mete foto del resultado a la apuesta y la mujer por ganada al Sandro.

Escenario 2

Juan y Miguel que va paseando por la calle ve como unos jóvenes hacen Parkour , un deporte muy técnico y físico que consta de dar saltos en la calle de forma que los obstáculos que hay en la vía desaparezcan o al menos se haga de una forma que parece fácil .

Juan y Miguel que tienen 16 años son aficionados, y Joan graba la forma en que hacen los saltos y sin decir nada a nadie envía un nuevo reto al Pedro, el que más fárrago del grupo de Parkour.

Como invitados pone a todo el grupo de Parkour y como jueces a las tres personas que él cree que más sabe del grupo.

Perfil: Juan y Miguel serán los administradores de la apuesta.

Objetivos: Crear una apuesta con más detalle gráfico y muchas más personas.

Tareas realizadas:

- Entrar en la aplicación.

- Crear la apuesta con la redacción de la apuesta, insertar el vídeo de ejemplo de lo que debe hacer, invitar a Pedro para que acepte la apuesta e invitar a todo el grupo.

- No pone fecha límite ya que no sabe cuándo podrán quedar todos para ver si lo puede hacer el Pedro.

- Los dos admins pueden realizar las modificaciones a la apuesta cuando quieran.

Información: Casualmente todo el grupo de Parkour está dispersa. Así que para quedar a una determinada meterán una fecha y hora para quedar para ver si Pedro después de tanto elogió a puede realizar las mismas piruetas que los del vídeo.

Desarrollo: Pedro deniega la apuesta, por lo tanto se da por finalizada la apuesta.

Pedro cree que no podrá realizar las pruebas, él es mucho hablar pero realmente sabe que tiene limitaciones, por lo tanto se la mofa del grupo y el chat se llena de vídeos de gente con caídas.

Escenario 3

La Maica y José van de camino a casa, como son estudiantes toman el metro y van juntos a mirar la tan maravillosa aplicación de apuestas, ven que han recibido varios mensajes de los amigos y quieren responder pero saben que no hay cobertura justo en el tramo que está ahora.

Aun así la conversación es tan interesante que no pudo dejarlo y escriben las gracias del resto de amigos.

Perfil: Ambos son observadores de la apuesta pero son muy activos en el grupo, siempre tienen algo que decir.

Objetivos: La aplicación debe saber guardar temporalmente los datos para así cuando haya cobertura poderlas enviar sin problemas.

Tareas realizadas: Enviar comentarios cuando no hay cobertura de datos.

Desarrollo: La pareja envía las respuestas pertinentes, la aplicación se las guarda y cuando vuelve a tener cobertura el móvil recibe los datos que el resto han dejado el chat y luego la aplicación envía los comentarios que se habían escrito.

Escenario 4

La misma pareja de antes, que son habituales de la aplicación, ni Maica y José, se encuentran otra vez en la misma tesitura que antes, de camino a casa, siguen siendo estudiantes pero esta vez vuelven en el coche con los papas.

Están planteando una nueva apuesta a su prima Laia que tiene un torneo de ajedrez y quieren motivar con una apuesta de por medio.

Así pues que empiezan a crear la apuesta pero esta vez justo en el medio de la creación reciben una llamada.

Perfil: Ambos son los administradores de la apuesta y Laia aunque no lo sabe será la que reten.

Objetivos: La aplicación debe saber guardar los datos dentro del formulario, tal y como ha quedado antes de la llamada para así poder continuar cuando finalice.

Tareas realizadas: Recuperar o mantener los datos en el formulario.

Desarrollo: La pareja envía y recupera la información sin ningún tipo de problema.

Escenario 5

El Sr. Mourinho que lleva un pique importante con el Sr. Guardiola y que ambos utilizan la aplicación quiere saber cuántas apuestas ha realizado en Guardiola de forma correcta.

Como todos sabemos que realmente son amigos cada uno tiene el teléfono del otro.

Son tan amigos que incluso se unieron a la apuesta del escenario 1 de Sandro y Florentino así que la rivalidad que ya no sienten en las ruedas de prensa, las pueden escuchar en la app...

Así pues quiere ver que dijo Guardilla en la apuesta pública del Barça -Madrid y también quiere ver si dentro del ranking de sus amigos él está por debajo de él.

Perfil: El Sr. Mourinho será un usuario normal y corriente de la aplicación.

Objetivos: Ver cuántas apuestas ha ganado y perdido en Guardilla, para poder así comparar, se mostrará en forma de ranking. También se realizará la búsqueda de una apuesta para ver cuál fue la resolución.

Tareas realizadas:

- Abrir la aplicación e ir al apartado de búsquedas de apuestas.

- Teclear sobre la búsqueda y pulsar Barça

- Aparecerán todas las apuestas que contengan Barça.

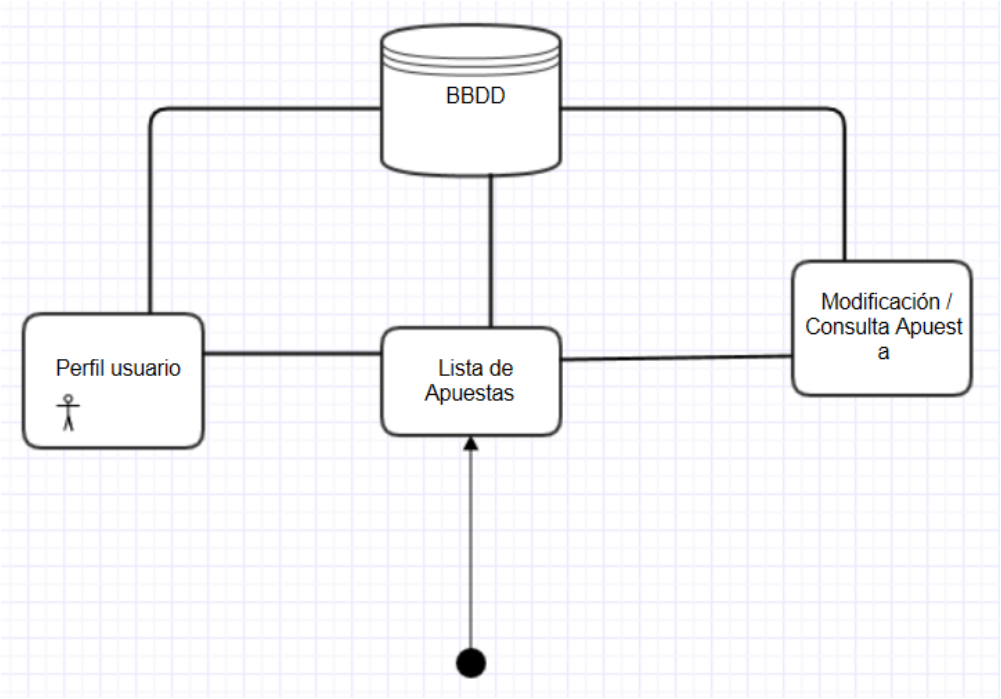
- Se vuelve al menú principal y pulsar sobre el Icono de ranking. Muestra las personas que más han acertado en proporción con las que menos han fallado.

Desarrollo: Tarea sencilla que se puede realizar de forma intuitiva. Puedes encontrar la apuesta que realizaste y recordar cuál fue su resolución. Buscará las palabras que contengan las palabras exactas. El ranking se realizará con todas las personas que hay en la aplicación.

Flujos de interacción.

Se presentará un diagrama que de manera gráfica muestre la estructura general de la aplicación.

Aunque no es la aplicación más acertada para este tipo de flujos que no queda clara del todo si que es una aplicación sencilla. Se ha utilizado Gliffy.



Prototipado de la aplicación [Diseño]

En este apartado se realizará una representación de la aplicación.

Permitirá comunicar decisiones de diseño y evaluar la aplicación o web móvil antes de desarrollarla.

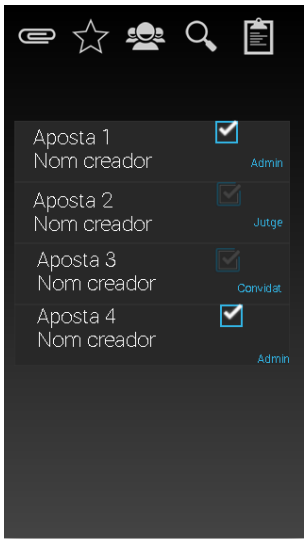
Su versatilidad hace que sea sencillo y económico introducir modificaciones en el diseño.

También es fácil iterar incorporando mejoras previa discusión con los miembros del equipo o de los resultados que se han obtenido en la evaluación.

Se visualizarán las diferentes pantallas de la aplicación con los elementos que contendrán.

Cuanto más precisas sean estas representaciones, los posibles comentarios que recibirá en la corrección podrán ser más útiles de cara a obtener una aplicación final más usable.

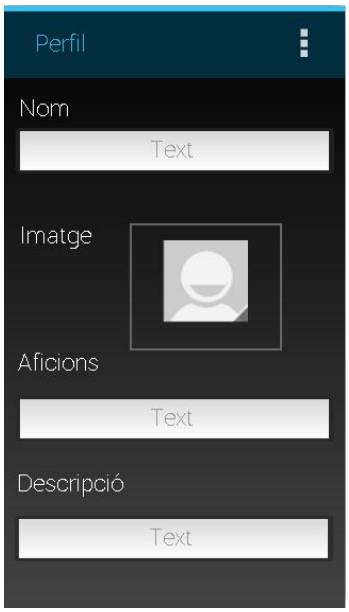
Es una aproximación de lo que se desea.



Después de iniciar la aplicación se mostrará un listado de las apuestas.

La pantalla de la apuesta tiene muchos campos y puede sufrir modificaciones estéticas aun así éste puede ser un primer diseño muy próximo al deseado.

Tanto la pantalla de creación como de modificación serán muy parecidas.



La pantalla de creación del perfil de usuario contiene el Nick, la descripción y la posibilidad de insertar una foto.

Implementación

Especificación Técnica del Proyecto

La versión para la que se va a realizar la aplicación será para la API 16 Android versión 4.1.2.

La Base de Datos donde se guardará la información es Sql Server 2012 Express.

El servidor web donde se publicarán los web services y hará de pasarela entre la aplicación y la BBDD será Apache Tomcat 6.0.32.

Todo este software requiere de una instalación y configuración para que trabaje de forma conjunta para darle funcionalidad a la aplicación.

Para que todo este software quede integrado de forma unitaria se utiliza Eclipse.

Este contendrá tres proyectos.

- El proyecto Android.
- El proyecto Webservice que accederá a la BBDD.
- El proyecto servidor Web para publicar el proyecto WebService y así acceder desde cualquier parte.

Instalación

Este apartado mostrará los pasos que se tienen que realizar para configurar todo el software utilizado.

Apache

Para poder publicar los webservice que conectan con la BBDD se necesita un Servidor Web.

En este caso escogemos Apache.

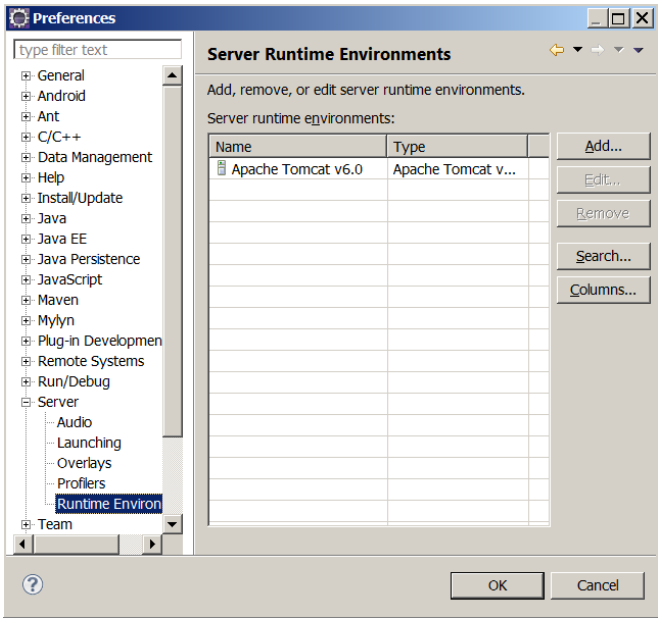
Lo bajamos de la página oficial: http://tomcat.apache.org

Realizados la descompresión del archivo en una carpeta común y realizamos la configuración de este servidor en Eclipse para poder publicar nuestros desarrollos.

Lo insertamos dentro del Eclipse.

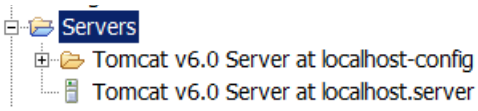
Para ello vamos a Windows->Preferences->Runtime Envorinment -> Botón Add

Seleccionamos el directorio de la instalación de Apache.

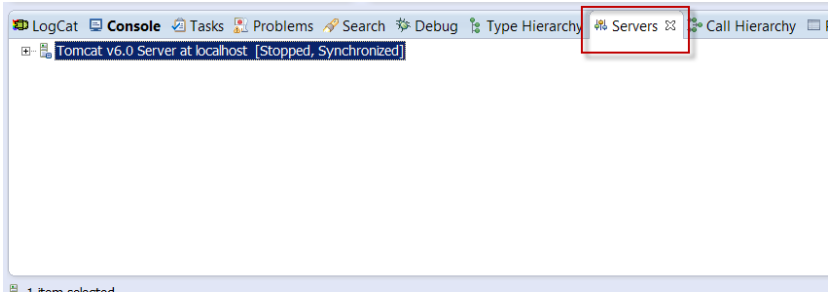


Apretamos en el botón Ok.

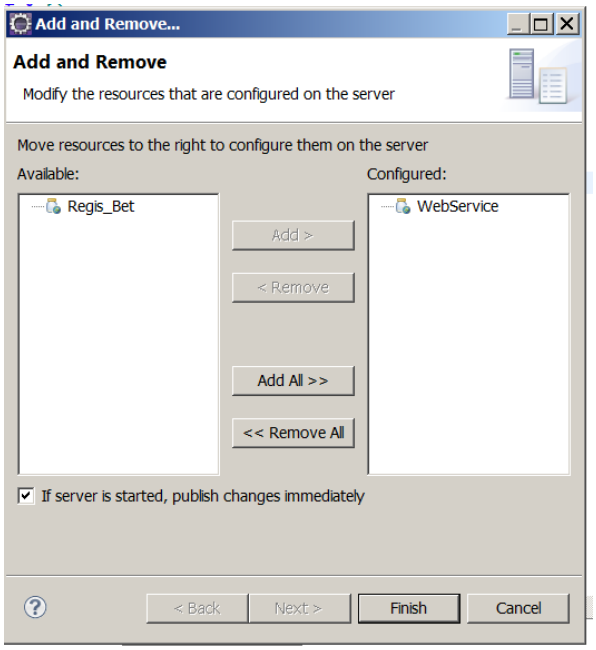
Aparecerá un proyecto nuevo Servers.



También aparecerá una nueva pestaña Servers donde se podrá añadir los proyectos que se van desarrollando para publicarlos.

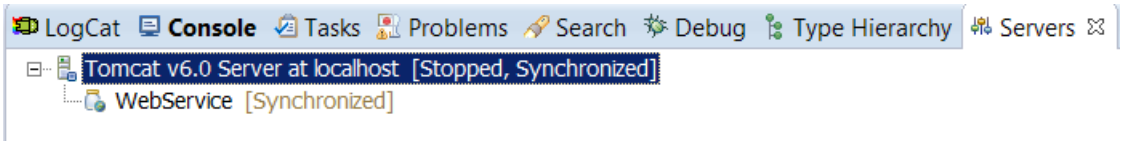


Para insertar y publicar el proyecto de servicios Web se debe clicar con el botón derecho sobre Tomcat -> Add and Remove. Aparecerá la siguiente pantalla.



Se escoge el proyecto en este caso WebService y se clica sobre add, aparecerá en la parte derecha que son los proyectos que se publicarán y se clica sobre Finish.

Aparecerá el proyecto seleccionado dentro del servidor apache.



Para que pueda soportar páginas estáticas estilo html cambiamos el fichero de configuración.

```
<param-value>false</param-value> a true.

<servlet>
        <servlet-name>default</servlet-name>
        <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-
class>
        <init-param>
            <param-name>debug</param-name>
            <param-value>0</param-value>
        </init-param>
        <init-param>
            <param-name>listings</param-name>
            <param-value>true</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
```

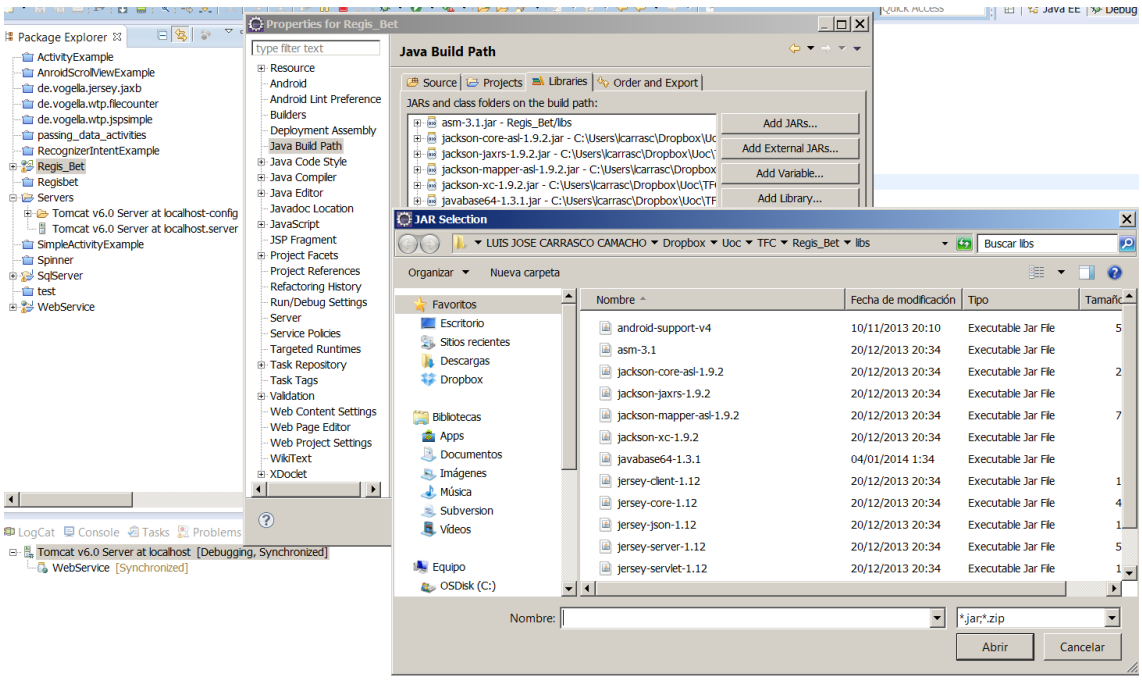
OW2

ASM es un marco para todo uso de Java bytecode manipulación y análisis. Se puede utilizar para modificar las clases existentes o dinámicamente generar clases, directamente en forma binaria. Transformaciones común siempre y algoritmos de análisis permiten montar fácilmente transformaciones personalizadas complejas y herramientas de análisis de código.

Nos vamos a la página de asm para bajarnos el plugin.

http://forge.ow2.org/project/download.php?group_id=23&file_id=19465

Download y lo insertamos dentro de WEB-INF lib.



Acto seguido botón derecho sobre el proyecto Android -> Properties-> Java Build Path-> Pestaña Libraries -> Add External JARs…

Aparece una nueva ventana se escoge el Jar asm y se clica Abrir.

Se verá en la pestaña de Libraries la nueva librería.

REstFul WebServices

Jersey y JSR 311

Se debe realizar lo mismo hacemos con jersey JSR 311.

Esta librería con la librería de Jersey (https://jersey.java.net/) te permite tener llamadas WebServices REST.

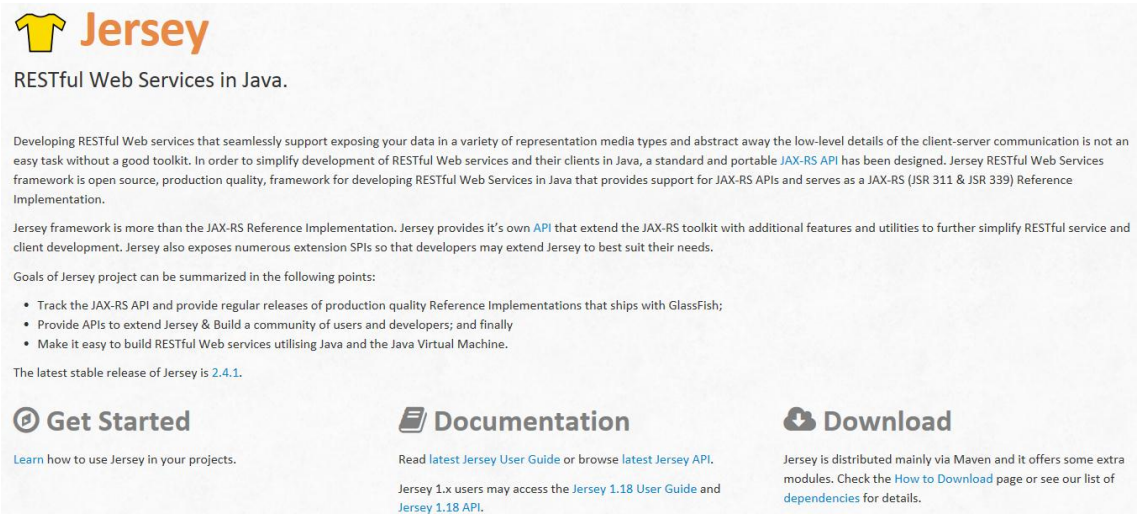
https://jsr311.java.net/

Para este caso nos bajamos la versión 1.1.

La descargamos y la insertamos dentro de la carpeta lib del proyecto, como se ha realizado en el apartado anterior.

Ahora le toca el turno a Jersey. Así que vamos a la página oficial para realizar la descarga:

https://jersey.java.net/



Con el fin de simplificar el desarrollo de servicios web RESTful y sus clientes en Java, un API JAX- RS estándar y portátil ha sido diseñado. La librería Jersey RESTful Web Services es de código abierto, de calidad suficiente para productos de producción, para el desarrollo de servicios web RESTful en Java que proporciona soporte para JAX -RS Apis y sirve como JAX-RS (JSR 311 y JSR 339)

Jersey ofrece su propia API que se extienden al conjunto de herramientas de JAX- RS con las características y utilidades adicionales para simplificar aún más el servicio RESTful y desarrollo de clientes.

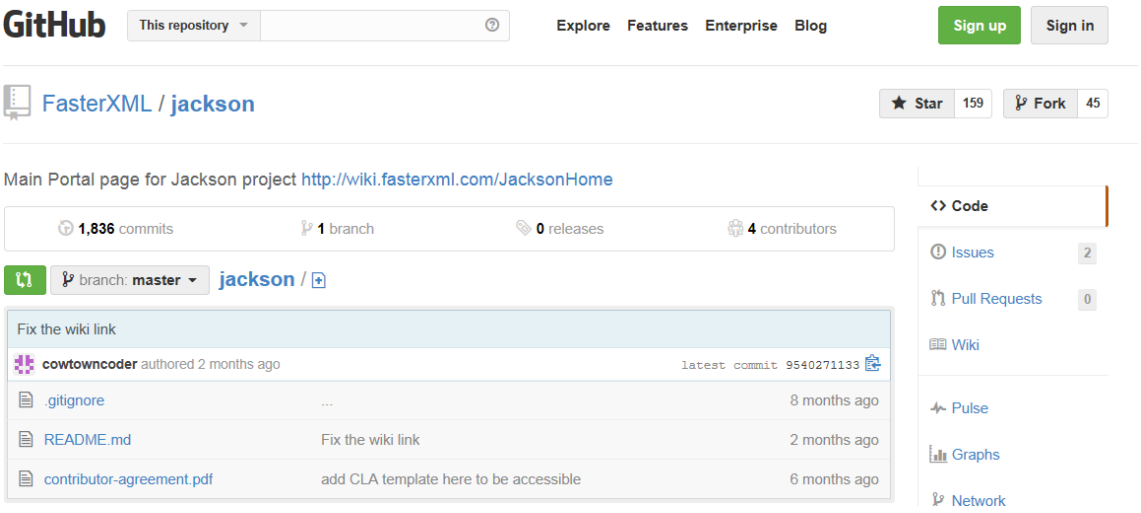
La versión actual es la 2.4.1. La descargamos y la insertamos dentro de la carpeta lib del proyecto.

Como finalmente no se ha podido realizar la configuración correcta para poder hacerla funcionar se utiliza la versión 1.12.

Jackson

Para el procesamiento de la información de los WebServices se utilizará Json con Jackson. Que según diversos estudios es una de las librerías más optimas existentes.

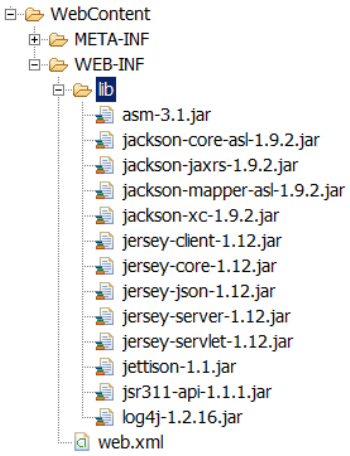
Para realizar la descarga vamos a la página https://github.com/FasterXML/jackson



Nos bajamos los jars y las insertamos dentro de la carpeta lib.

Tras haber invertido alrededor de un día para intentar configurar la versión más actual se ha procedido a bajar versiones anteriores que en este caso funciona.

Por tanto el listado de jars que he conseguido hacer funcionar, con sus versiones, se muestran en la imagen:



En el proyecto de webService se modifica el fichero Web.xml para poder dar soporte a las llamadas Restfull.

El Web.xml de configuración para el Web Service RESTful queda:

```
<display-name>BBDD and RESTfull WebService</display-name>

 <servlet>

    <servlet-name>RESTfull WebService</servlet-name>
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
     <init-param>
```

```
    <param-name>com.sun.jersey.config.property.packages</param-name>
    <param-value>com.suryasuravarapu.jersey</param-value>
  </init-param>

  <init-param>
    <param-name>com.sun.jersey.api.json.POJOMappingFeature</param-name>
    <param-value>true</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>

  <servlet-name>RESTfull WebService</servlet-name>
  <url-pattern>/*</url-pattern>
</servlet-mapping>
```

Base de Datos (BBDD)

Instalación BBDD

Nos bajamos la BBDD Microsoft Sql Server 2012 Express de http://www.microsoft.com/en-us/download/details.aspx?id=29062.

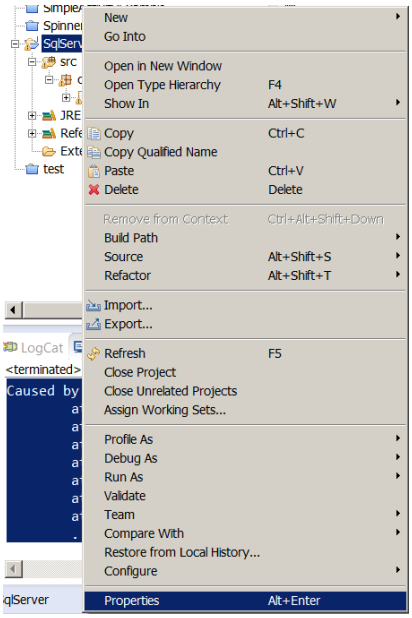
La instalación de la BBDD es clicando sobre el botón de siguiente se rellena el nombre del servidor y la seguridad que se le quiere aplicar.

En este caso se coge la seguridad integrada en la que el sistema es el integrado con el sistema operativo del servidor que en este caso es Windows 7.

Conexión BBDD con Eclipse

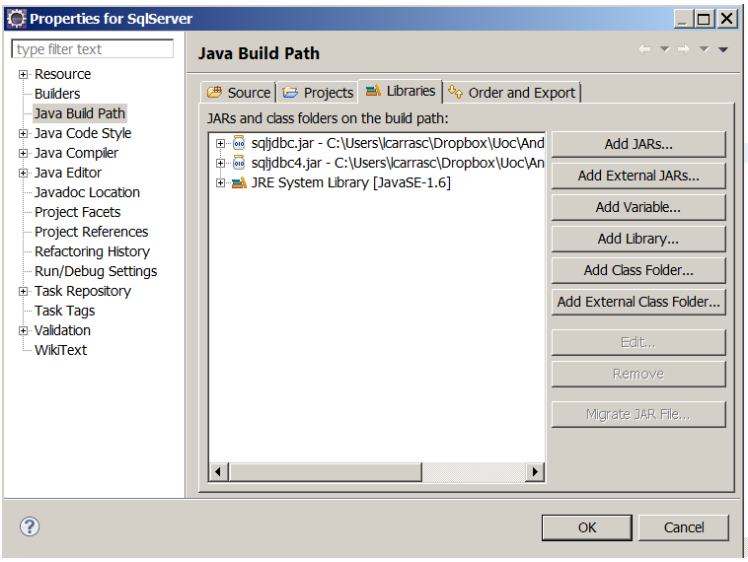
Nos bajamos los Jar de jdbc de conexión para eclipse con la BBDD en http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774.

Insertamos dentro del Proyecto que va a tener la conexión (WebService) y el control con la BBDD las librerías Jar descargadas.



Propiedades en el proyecto.

Aparece una pantalla nueva. Se escoge java Build Path, pestaña librerías y mediante el botón "Add External JARs" se insertan los Jar para la conexión a la BBDD.

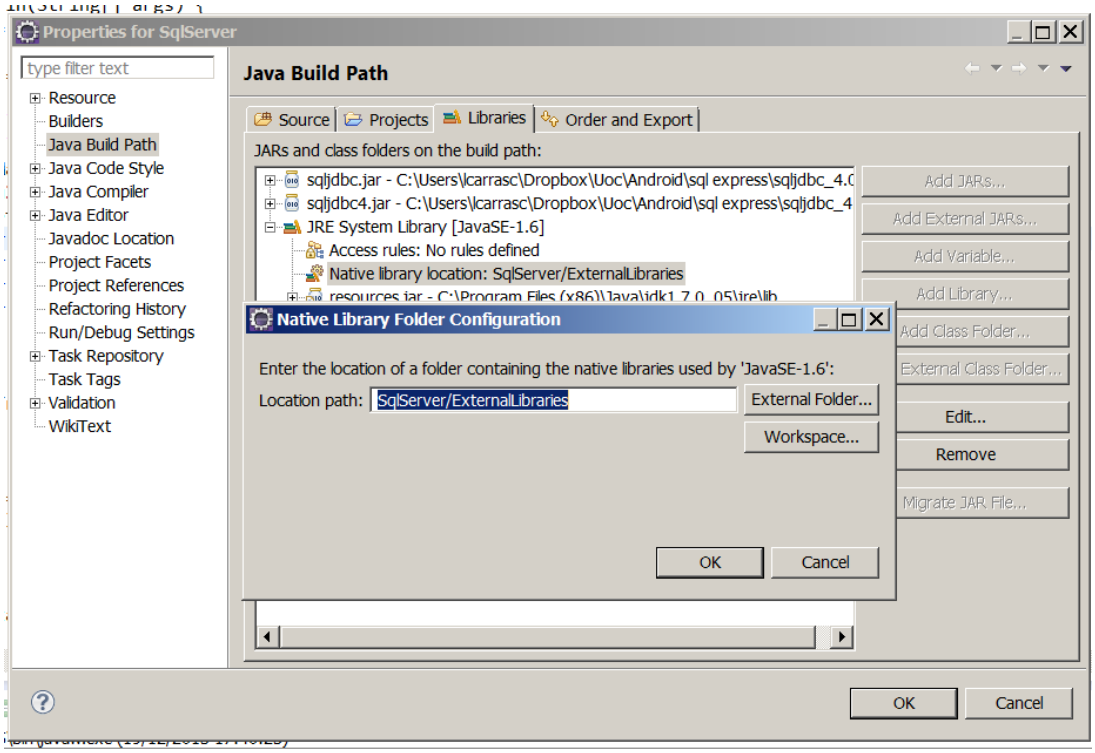


Dentro del proyecto creamos una clase de prueba de conexión y nos muestra el siguiente error:

```
dic 19, 2013 5:40:28 PM com.microsoft.sqlserver.jdbc.AuthenticationJNI <clinit>
Advertencia: Failed to load the sqljdbc_auth.dll cause : no sqljdbc_auth in java.library.path
com.microsoft.sqlserver.jdbc.SQLServerException: Este controlador no está configurado para la autenticación
integrada. ClientConnectionId:ce1159b9-75b4-40c7-bd05-60d129703959
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.terminate(SQLServerConnection.java:1667)
        at com.microsoft.sqlserver.jdbc.AuthenticationJNI.<init>(AuthenticationJNI.java:60)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.logon(SQLServerConnection.java:2229)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.access$000(SQLServerConnection.java:41)
        at
com.microsoft.sqlserver.jdbc.SQLServerConnection$LogonCommand.doExecute(SQLServerConnection.java:2220)
        at com.microsoft.sqlserver.jdbc.TDSCommand.execute(IOBuffer.java:5696)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.executeCommand(SQLServerConnection.java:1715)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.connectHelper(SQLServerConnection.java:1326)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.login(SQLServerConnection.java:991)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.connect(SQLServerConnection.java:827)
        at com.microsoft.sqlserver.jdbc.SQLServerDriver.connect(SQLServerDriver.java:1012)
        at java.sql.DriverManager.getConnection(DriverManager.java:579)
        at java.sql.DriverManager.getConnection(DriverManager.java:243)
        at com.sqlServer.Connect.Consulta.main(Consulta.java:16)
Caused by: java.lang.UnsatisfiedLinkError: no sqljdbc_auth in java.library.path
        at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1860)
        at java.lang.Runtime.loadLibrary0(Runtime.java:845)
        at java.lang.System.loadLibrary(System.java:1084)
        at com.microsoft.sqlserver.jdbc.AuthenticationJNI.<clinit>(AuthenticationJNI.java:35)
        ... 12 more
```

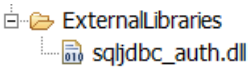
Después de haber mirado por internet encontramos que es debido a la autentificación.

Con la lectura de estos dos links que tenían una problemática similar se ha llegado a la conclusión de que se debe copiar una dll de autentificación de Sql server en el Path de `java.library.path.` Para ello se debe copiar el dll de autentificación que hay en el directorio de jdbc.



Para ello volvemos al apartado de “Java Build Path” pestaña Libraries apartado “JRE System Library” apartado “Native library Location” se clica sobre el botón “Edit…”. En nuestro caso se pone el Path del Workspace y se crea una nueva carpeta “External Libraries”.

Cuando se ha creado se copia la .dll dentro de la carpeta.



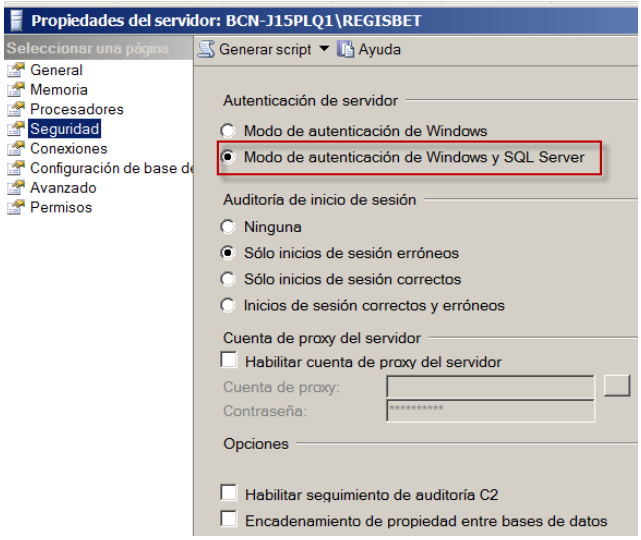
Se vuelve a probar y el resultado es el siguiente:

```
Driver name: Microsoft JDBC Driver 4.0 for SQL Server
Driver version: 4.0.2206.100
Product name: Microsoft SQL Server
Product version: 11.00.2100
```

El resultado es propio pero indica que se ha podido establecer la llamada a la BBDD correctamente.

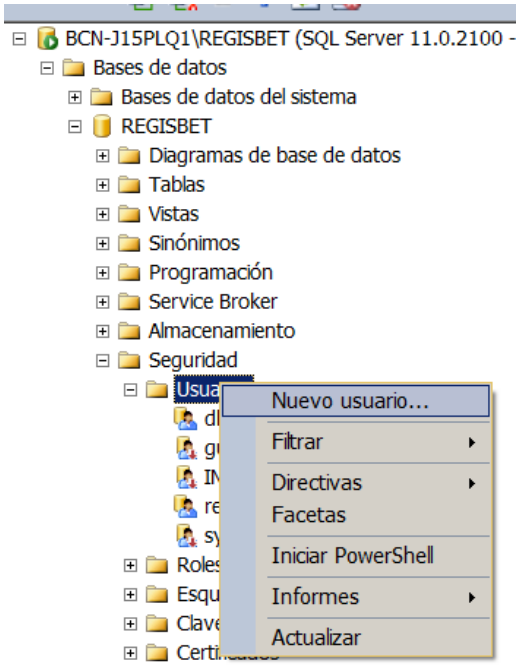
Aun así se encontró un problema y es que al haber escogido el login de forma integrada en la BBDD cuando la petición se realizaba desde la aplicación no era posible la conexión a la base de datos.

Por ello se tuvo que cambiar el sistema de login a integrada de Windows y SQL Server. Para ello clicamos sobre el servidor con el botón derecho-> Propiedades -> Seguridad.



Como no había usuario, se tuvo que crear uno.

Botón derecho en el apartado de usuarios -> Nuevo usuario.

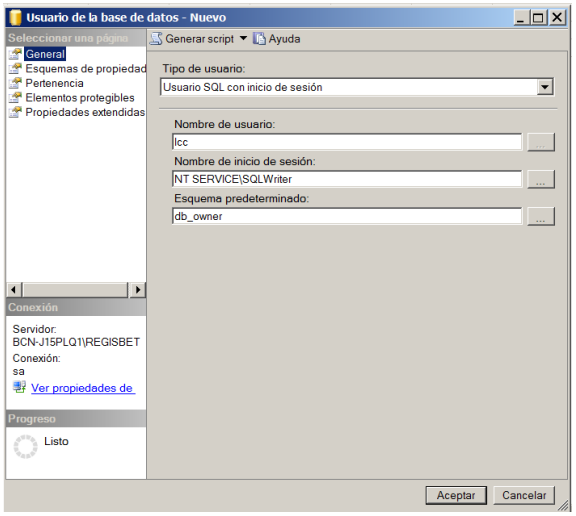


Rellenamos los campos que nos piden y ya está listo para poder realizar el login de nuevo.

En este caso se ha escogido el nombre.

El más importante es de darle los privilegios de propietario de la BBDD con el apartado Esquema predeterminado.

También se ha habilitado el usuario sa.



Con estas modificaciones se realiza la petición desde la aplicación móvil y volvemos a tener el mismo mensaje de respuesta:

```
Driver name: Microsoft JDBC Driver 4.0 for SQL Server
Driver version: 4.0.2206.100
Product name: Microsoft SQL Server
Product version: 11.00.2100
```

Descarte de software y plugins

Este apartado indicamos cuales han sido los plug-ins de Eclipse que se han intentado instalar y que por motivos diversos no se han podido finalmente utilizar.

Maven

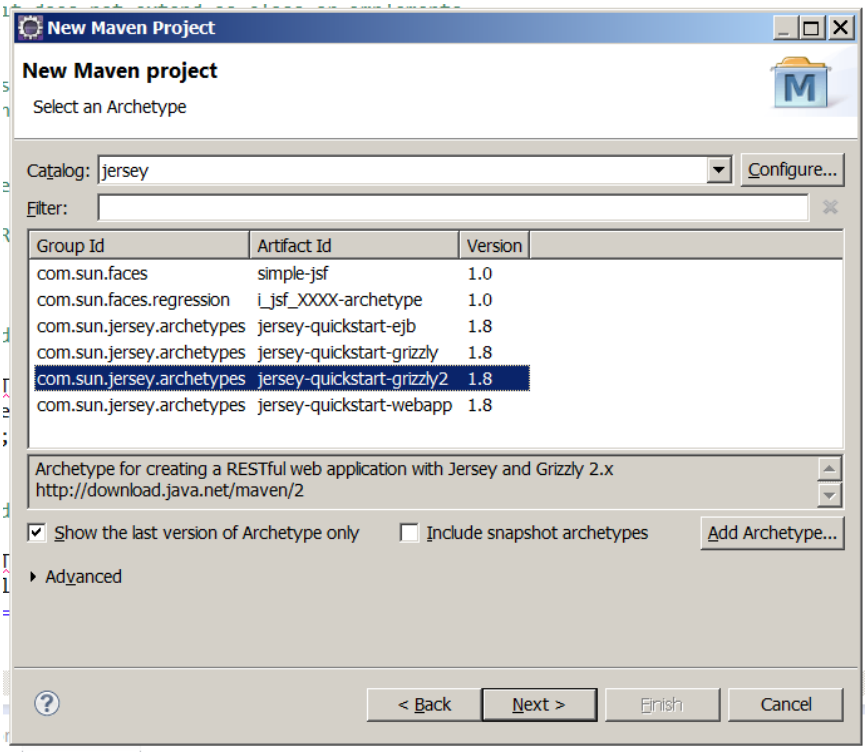
http://maven.apache.org/

Lo pensaba utilizar por que puede facilitar la descarga de los requisitos de un proyecto de forma automática y simplifica todo bastante.

Para empezar con el proyecto se debe crear el archetype catalog para que lo cree.

http://download.java.net/maven/2/archetype-catalog.xml

Para ello creamos un Nuevo proyecto tipo maven:



Elegimos jersey-quickstart-webapp y presionamos sobre enter y finish.

Después de un par de horas de haber intentado hacerlo funcionar lo descarto por que no he conseguido descargar nada y creo que de forma manual puedo llegar a saber que es lo que estoy haciendo en todo momento.

Conclusión

Se descarta por que no he encontrado como hacer para bajarme los proyectos de las diferentes páginas.

No se si es por que no realizo bien la petición, vía comando o por eclipse pero se le ha dedicado demasiado tiempo para no obtener resultado alguno.

Jersey 2.4.1

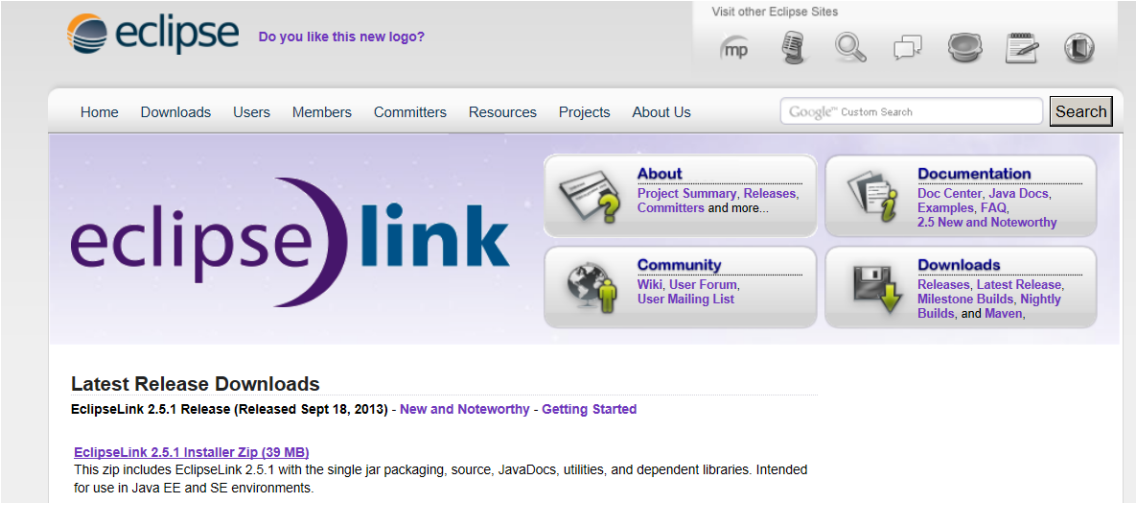
La versión más actualizada que hay actualmente en la web no he sabido hacerla funcionar.

Me he tenido que conformar con bajarme una versión muy anterior para poder hacer peticiones tipo REST.

He visto que el XML de configuración es diferente a la que he podido hacer funcionar, pero debe de haber otras variables que no he encontrado, por haberle dedicado tiempo en exceso y viendo que la versión anterior funcionaba se mantiene la anterior.

Persistencia Datos Eclipse Link

Para mantener la relación de los datos con la BBDD y la comunicación entre los diferentes WebServices se va a utilizar Eclipse Link.



Bajamos el instalador de:

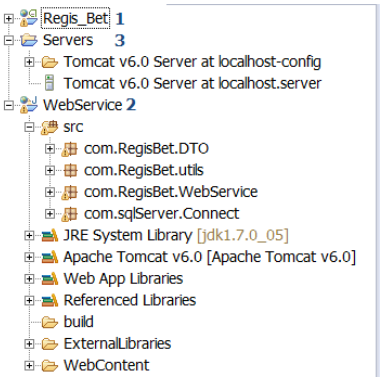
http://www.eclipse.org/downloads/download.php?file=/rt/eclipselink/releases/2.5.1/eclipselink-2.5.1.v20130918-f2b9fc5.zip

Pero no he sabido como configurarlo para que con la conexión de jdbc y Sql server aparezcan las opciones de persistencia, así que se descarta.

Fase de implementación RegisBet

En este apartado se va a comentar en detalle cual es la implementación llevada a cabo.

Como se ha comentado con anterioridad se crean tres proyectos diferenciados dentro del IDE Eclipse.



1- Regis_Bet: Proyecto Android donde estará ubicada toda la lógica de la aplicación móvil.

2- WebService: Servicios Rest para comunicación entre BBDD y aplicación. Publicados mediante Apache Tomcat del tercer proyecto.

3- Server: Proyecto creado para la publicación de proyectos.

A continuación se realizará una descripción detallada de los proyectos ordenada de menos a más relevante.

Proyecto Servers

Para poder acceder a los datos desde cualquier parte del mundo debe de haber un servidor web.

Este proyecto tiene la finalidad de emular como se comportaría el servidor web Apache Tomcat con la misma configuración.

Con este proyecto publicamos todo lo que se va desarrollando, sin ningún tipo de configuración extra.

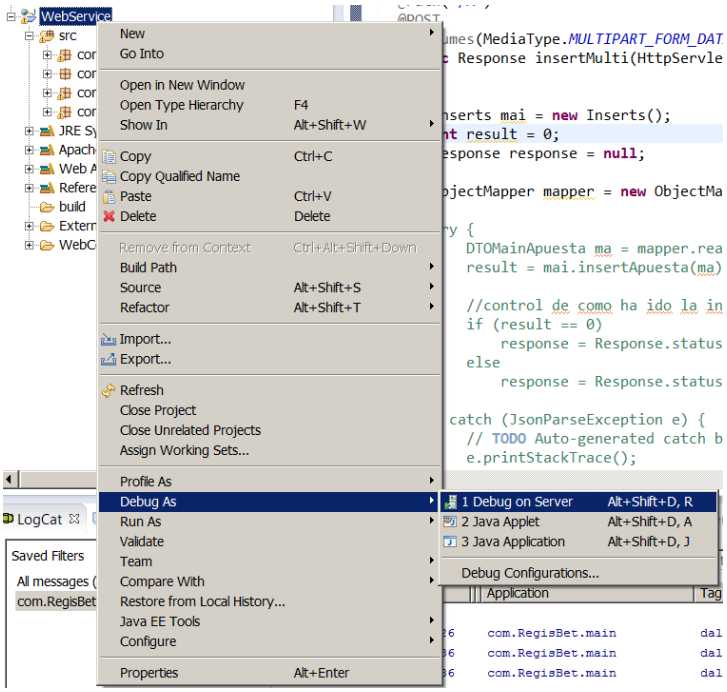
Gracias a este proyecto se nos permite realizar la traza de las peticiones desde la aplicación al servidor en modo debug o lo que es lo mismo, poder ver que se está ejecutando en cada momento.

Para la publicación de los proyectos se puede hacer de diversas formas.

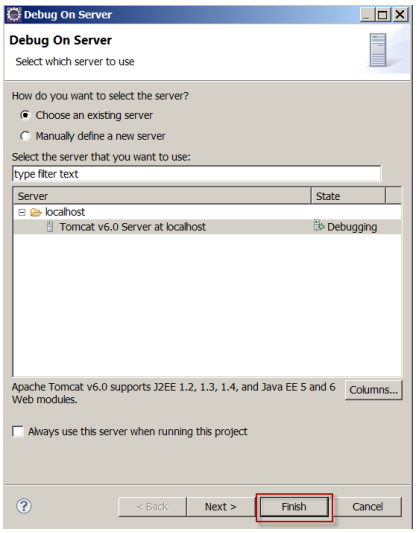
Publicar en el Servidor

Desde el proyecto que se quiere publicar se indica que quieres hacerlo en los servidores configurados. En nuestro caso tenemos uno.

Para hacerlo funcionar es tan sencillo como clicar sobre botón derecho en el proyecto Webservice-> Debug As-> 1Debug on Server.



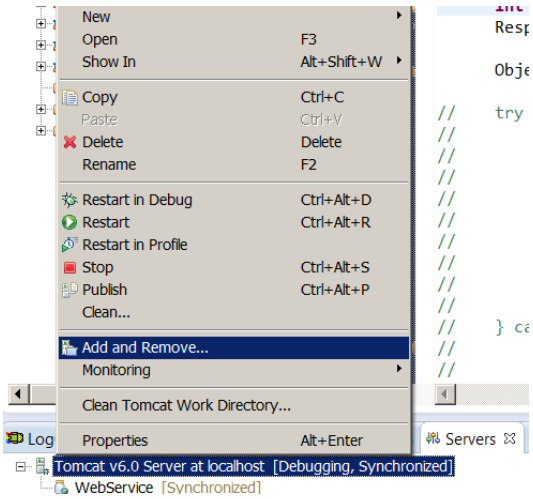
Aparecerá la pantalla para confirmar en que servidor Web se quiere publicar, en nuestro caso solo hay uno, así que la decisión es sencilla. Se clica sobre Finish para publicar



Cuando finaliza se abre una pantalla mostrando lo publicado en el servidor. En mi caso no tengo nada así que se muestra lo siguiente:

Pestaña Servers

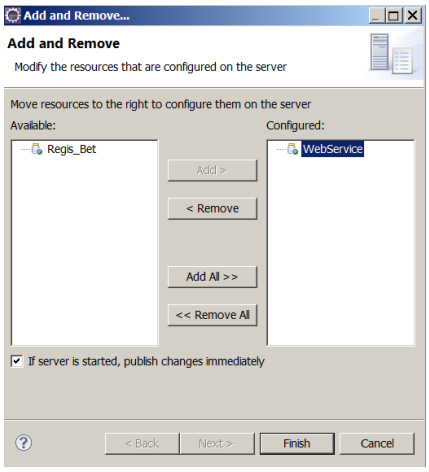
Otra forma de publicar un proyecto es desde la pestaña de Servers.



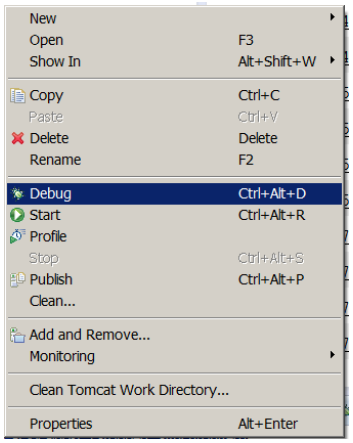
Como se muestra en la imagen aparecen los servidores Web que hay configurados.

Para poder publicarlo se debe clicar con el botón derecho sobre “Tomcat v6.0 Server at localhost” -> Add and Remove.

Aparece la siguiente pantalla indicando cuales son los proyectos que se quieren publicar se ponen a la derecha y se clica sobre Finish.



Después de esto se debe de iniciar el servidor Web.



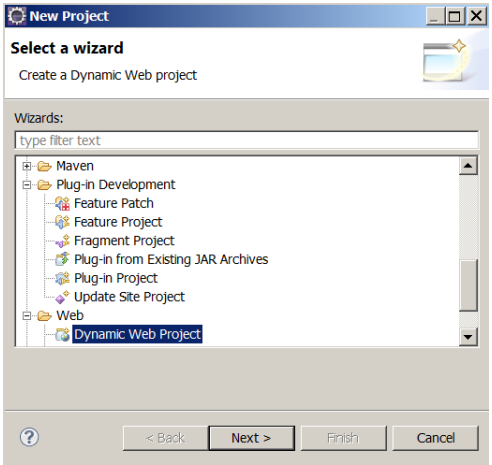
En desarrollo es muy conveniente reiniciarlo en Debug ya que te permite detener el código en el punto deseado con los Breakpoints.

Proyecto WebService

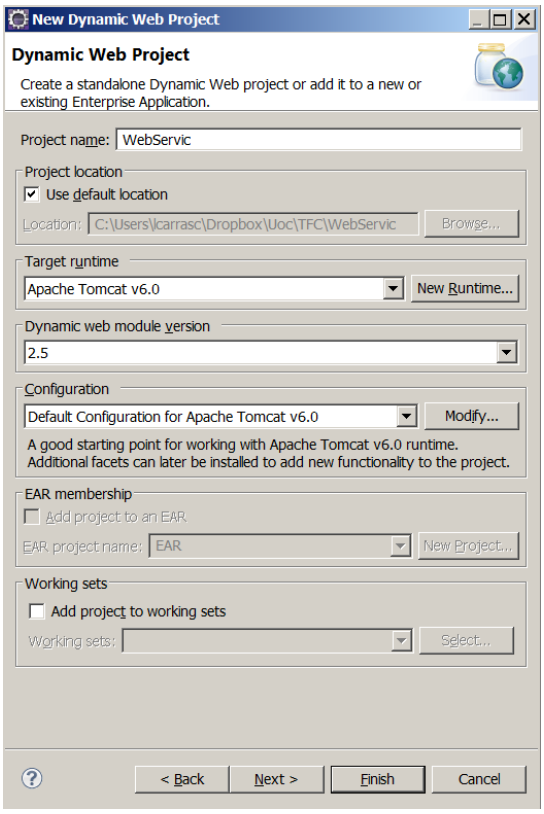
Este proyecto contiene la lógica de conexión e inserción de datos a la BBDD.

Es un proyecto “Dynamic Web Project” que se crea con File->New->Project.

Aparece la siguiente ventana, donde se debe elegir que tipo de proyecto se quiere crear. Se clica sobre Next.



Aparece la pantalla donde tienes que indicar que nombre le quieres poner, en que servidor Web quieres hacerlo funcionar y si quieres que tenga alguna configuración especial.



Para este proyecto se ha rellenado en nombre del proyecto y se ha clicado sobre finish.

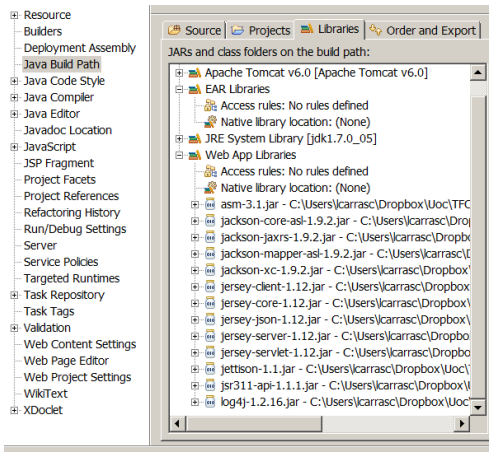
Una vez creado se debe desarrollar las clases que reciben los datos.

En nuestro caso hemos escogido que se realizará por web Service tipo REST, encapsulado mediante Json.

Con estos datos los transforma en objetos propietarios, extrae los datos necesarios, realiza la conexión a la BBDD, inserta los datos y devuelve el estado de como ha ido el desarrollo de la sentencia insert, select o Update.

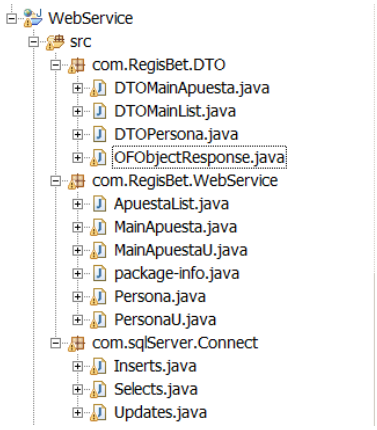
Este proyecto también realiza la inserción de las imágenes recibidas en el servidor para que se puedan recoger desde la app.

Las librerías utilizadas son las siguientes:



1) Asm: es un marco para manipulación y análisis de bytecode. Se utiliza para modificar las clases existentes o dinámicamente generar clases, directamente en forma binaria.
2) Jackson: librería de Json para la transformación de los datos enviados en Json.
3) Jersey: librería para la abstracción de los datos enviados vía REST de Json a clases propias de forma sencilla.
4) Jsr311: Librería complementaría y necesaria para la utilización de Jersey.
5) Log4j: API para realizar trazas del log de forma más exhaustiva y sencilla en el servidor Web.
6) Jettison: librería complementaria para JSON.

Con estas librerías se ha generado el proyecto con la siguiente estructura de código:



Package com.RegisBet.DTO

Contiene las clases que hacen referencia a los datos de la tabla. Por ejemplo la clase DTOMainApuesta.java contiene todos los datos recibidos para poder mostrar y guardar los datos. En cambio la clase DTOMainList.java es la clase encargada de representar la información que aparecerá en la pantalla principal mostrando las apuestas vigentes.

A continuación se muestra parte del código de DTOMainApuesta.java ya que es la más compuesta y representativa de todas:

DTOMainApuesta.java

```
public class DTOMainApuesta {

        private String ApuestaId = "";
        private String TituloApuesta = "";
        private Date FechaFin;
        private String Descripcion = "";
        private String RespuestaId = "";
        private String RecuentoId = "";
        private String ListadoArchivo = "";
        private String XatId = "";
        private String Estado = "";
        //New Version
        private boolean privado = false;
        private String imagen = "";
        private byte[] imagenB = null;
        private String Video = "";
        private NavigableMap<String, String> respuestaValues = new TreeMap<String, String>();
        private String imagenTh = "";
        private byte[] imagenBTh = null;
        private String Owner = "";
        private HashMap<String, String> ListadoPersonaId = new HashMap<String, String>();
        private String personaContra = "";

        public DTOMainApuesta() {

        }

        /**
         * @return the apuestaId
         */
        public String getApuestaId() {
                return ApuestaId;
        }
        /**
         * @param apuestaId the apuestaId to set
         */
        public void setApuestaId(String apuestaId) {
                ApuestaId = apuestaId;
        }
        /**
         * @return the tituloApuesta
         */
        public String getTituloApuesta() {
                return TituloApuesta;
        }
        /**
         * @param tituloApuesta the tituloApuesta to set
         */
        public void setTituloApuesta(String tituloApuesta) {
                TituloApuesta = tituloApuesta;
        }
        /**
         * @return the fechaFin
         */
        public Date getFechaFin() {
                return FechaFin;
        }
        /**
         * @param parsedDate the fechaFin to set
         */
        public void setFechaFin(Date parsedDate) {
                FechaFin = parsedDate;
        }
        /**
         * @return the descripcion
         */
        public String getDescripcion() {
                return Descripcion;
        }
        /**
         * @param descripcion the descripcion to set
```


```
        ……. …… ……. …… …..
        ……. …… ……. …… …..
        ……. …… ……. …… …..

        /**
         * @return the listadoPersonaId
         */
        public HashMap<String, String> getListadoPersonaId() {
                return ListadoPersonaId;
        }
        /**
         * @param listadoPersonaId the listadoPersonaId to set
         */
        public void setListadoPersonaId(HashMap<String, String> listadoPersonaId) {
                ListadoPersonaId = listadoPersonaId;
        }
        /**
         * @return the listadoArchivo
         */
        public String getListadoArchivo() {
                return ListadoArchivo;
        }
        /**
         * @param listadoArchivo the listadoArchivo to set
         */
        public void setListadoArchivo(String listadoArchivo) {
                ListadoArchivo = listadoArchivo;
        }       /**
         * @param personaContra the personaContra to set
         */
        public void setPersonaContra(String personaContra) {
                this.personaContra = personaContra;
        }

        /**
         * @return the imagenB
         */
        public byte[] getImagenB() {
                return imagenB;
        }

        /**
         * @param imagenB the imagenB to set
         */
        public void setImagenB(byte[] imagenB) {
                this.imagenB = imagenB;
        }

        /**
         * @return the imagenBTh
         */
        public byte[] getImagenBTh() {
                return imagenBTh;
        }

        /**
         * @param imagenBTh the imagenBTh to set
         */
        public void setImagenBTh(byte[] imagenBTh) {
                this.imagenBTh = imagenBTh;
        }

}
```

La explicación de algunos campos es la siguiente:

- ApuestaId: Contiene el número identificador de la apuesta, así también como el número identificador del listado de personas contra quien apuestas y las respuestas posibles que se dan.
- respuestaValues: de tipo NavigableMap que contiene el valor de todas las respuestas de la apuesta.
- imagenB: Byte[] que contiene toda la imagen.
- Imagen: String con nombre de la imagen.
- Owner: Creador de quien hace la apuesta.
- ListadoPersonaId: HashMap<String, String> que representa el listado de personas contra quien va dirigida la apuesta.
- personaContra: contiene el id que se guardará en la BBDD, que es el mismo que ApuestaId.

Como se ve en la imagen anterior también hay una clase que está para la representación/transformación de la información.

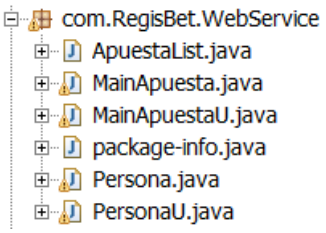
OFObjectResponse

```
public class OFObjectResponse {

        private int error = 0;
        private String response = "";
        private List<Object> betList;

        public OFObjectResponse(int error, String response) {
                setError(error);
                setResponse(response);
        }

        /**
         * @return the error
         */
        public int getError() {
                return error;
        }

        /**
         * @param error the error to set
         */
        public void setError(int error) {
                this.error = error;
        }

        /**
         * @return the response
         */
        public String getResponse() {
                return response;
        }

        /**
         * @param response the response to set
         */
        public void setResponse(String response) {
                this.response = response;
        }

        /**
         * @return the betList
         */
        public List<Object> getBetList() {
                return betList;
        }

        /**
         * @param betList the betList to set
         */
        public void setBetList(List<Object> betList) {
                this.betList = betList;
        }
```


Esta clase está representada por dos partes.

- `Error: cuando tiene valor 0 o -1 es que ha habido un error en la recuperación / inserción o update de los datos.`
- `Response: contiene serializado los datos de envío a la aplicación.`

Package com.RegisBet.WebService



Este package se encarga de recibir las llamadas tipo Rest de la aplicación. Únicamente se ha utilizado las llamadas tipo @GET y @POST. Es cierto que para la inserción de las imágenes se podría haber utilizado el @PUT, pero como también hay realización de insert en la BBDD se ha utilizado @POST.

Las clases tienen una función unitaria.

Por lo tanto la clase ApuestaList.java se encarga únicamente de realizar las select del listado de apuestas de la pantalla principal.

- MainApuesta.java se encarga de realizar el select e insert en la BBDD de la apuesta, con sus respectivas opciones de respuestas y contra quien se realiza.
- MainApuestaU.java realiza el Update de las tablas de la apuesta controlando también las respuestas y contra quien se realiza la apuesta.
- Persona.java controla las acciones de inserción y recuperación de datos de la persona.
- PersonaU.java realiza los updates en la modificación de los datos de la persona.

A continuación vamos a comentar el código de Persona.java para ver como funciona:

Persona.java

```
@GET
 @Path("{idUsuario}")
 @Produces(MediaType.APPLICATION_JSON + ";charset=utf-8")
public OFObjectResponse getPersona(@PathParam("idUsuario") String idUsuario) {

        try {
                Selects mas = new Selects();

                  DTOPersona persona = mas.selectPersona(idUsuario);

                  if (persona == null)
                          return new OFObjectResponse(-1,"El usuario no existe");
                  else {
                          ObjectMapper mapper = new ObjectMapper();
                          String personaString = new String();
                          personaString =  mapper.writeValueAsString(persona);

                          return new OFObjectResponse(0, personaString);
                  }

        } catch (Exception e) {
                 Logger.getLogger(getClass()).error(e.getMessage());
                 return new OFObjectResponse(-1,e.getMessage());
         } finally {

         }

 }
```

Función que recibe por parámetro el idUsuario del que recuperar la información.

Esta función se encarga de recoger la información de la BBDD mediante la llamada a mas.selectPersona(idusuario) la guarda en la clase DTOPersona, la serializa a Json mediante una función de Jackson y mediante el objeto OFObjectResponse comentado anteriormente se envía de vuelta.

El código que devuelve es de tipo Json y eso debe quedar especificado en la función con la línea: `@Produces(MediaType.APPLICATION_JSON + ";charset=utf-8")`. Se inserta ";charset=utf-8" para que tenga soporte en UTF-8 si se quita este apartado, por ejemplo no puede representar correctamente los acentos y caracteres especiales.

```
@POST
  @Consumes(MediaType.APPLICATION_JSON)
 @Produces (MediaType.APPLICATION_JSON)
  public OFObjectResponse create(String persona) {

          Inserts mai = new Inserts();
           int result = 0;
           Response response = null;

           ObjectMapper mapper = new ObjectMapper();

           try {
                    DTOPersona ma = mapper.readValue(persona, DTOPersona.class);
                    result = mai.insertPersona(ma);

                    if (result == 0){
                              return new OFObjectResponse(-1,"Problemas al guardar Persona");

                    }
                    else {
                              return new OFObjectResponse(0, "Operación realizada con éxito");

                    }

           } catch (JsonParseException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           } catch (JsonMappingException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           }
         return new OFObjectResponse(-1,"Problemas al guardar Persona");

 }
```

Para la función créate la acción que realiza es la de inserción en BBDD con la llamada a `mai.insertPersona(ma);`

En caso que haya problemas esta función devolverá 0 si todo ha ido bien 1.

PersonaU.java

```
@POST
 @Consumes(MediaType.APPLICATION_JSON)
 @Produces (MediaType.APPLICATION_JSON)
  public OFObjectResponse create(String persona) {

          Updates mau = new Updates();
           int result = 0;
           Response response = null;

           ObjectMapper mapper = new ObjectMapper();

           try {
                    DTOPersona ma = mapper.readValue(persona, DTOPersona.class);
                    result = mau.updatePersona(ma);

                    if (result == 0){
                             return new OFObjectResponse(-1,"Problemas al guardar Persona");

                    }
                    else {
                             return new OFObjectResponse(2, "Operación realizada con éxito");

                    }

           } catch (JsonParseException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           } catch (JsonMappingException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
           }
         return new OFObjectResponse(-1,"Problemas al guardar Persona");
```

Función muy similar a la anterior pero en esta ocasión se encarga de realizar el Update con la llamada a `result = mau.updatePersona(ma);` donde mau es de tipo Update que es una clase propietaria.

Package com.sqlServer.Connect

Package que tiene la misión de realizar la conexión de la BBDD, recoger los valores, realizar la acción pertinente (insert, Update y select) y devolver los valores.

El insert más completo es el de la apuesta principal.

Insert.java

```
    /***
     * Realiza el insert dentro de la BBDD en la tabla MainApuestas y todas aquellas que estan
intercomunicadas.
     *
     * @param ma Clase representativas de La apuesta
     * @return 1 si ha ido bien 0 sino ha isertado nada.
     */
    public int insertApuesta(DTOMainApuesta ma) {

        int result = 0;
        Connection conn = null;

        try {
                String connexion = configFile.getProperty("connexion");
                connexion = connectString;

                conn = DriverManager.getConnection(connexion);
                if (conn != null) {
                        //Reamos el Id de la apuesta que tambien sera el de las respuestas
                        double random = Math.random();
                        String id = String.valueOf(random).substring(String.valueOf(random).length()-
10,String.valueOf(random).length());

                        //Insertamos las respuestas posibles de la apuesta
                        int insertRespuestas = insertRespuesta(conn, id, ma.getRespuestaValues());
                        int insertContras = insertContra(conn, id, ma.getListadoPersonaId());

                        if (insertRespuestas == 1 && insertContras == 1) {

                                saveImages(ma, id);
                                String sentencia = createInsertSentence(ma, id);
                                PreparedStatement stmt = conn.prepareStatement(sentencia);
                        result = stmt.executeUpdate();

                        stmt.close();
                        conn.close();
                        }
                        else return 0;

                }
        } catch (FileNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        catch (IllegalArgumentException ex) {
                ex.printStackTrace();
            } catch (SQLException ex) {
                ex.printStackTrace();
            } finally {
                try {
                        if (conn != null && !conn.isClosed()) {
                                conn.close();
                        }
                }
                catch (SQLException ex) {
                        ex.printStackTrace();
                }
        }

        return result;
    }
```

Los pasos que se realizan para insertar los datos de la apuesta en la BBDD son los siguientes:

- Conseguimos los datos de conexión y abrimos la conexión `conn = DriverManager.getConnection(connexion);`

- Creamos el identificador de la apuesta con la función random.
- Insertamos las respuestas y el listado de contra quien va dirigida la apuesta.
- Si todo ha ido correctamente se procede a la inserción de la apuesta.
- Para ello primero se graban las imágenes las imágenes.
- Se crea la sentencia que va a ejecutarse en la BBDD.
- Se ejecuta con el comando `PreparedStatement stmt = conn.prepareStatement(sentencia);`
  `result = stmt.executeUpdate();`

Proyecto Android Regis_Bet

En este proyecto esta ubicada toda la lógica y diseño que el usuario dispondrá en su terminal móvil para poder realizar las apuestas.

Se realiza en Android que es el nombre con el que se denomina al SO para dispositivos móviles desarrollado por la una alianza de empresas comandada por Google y en las que participan organizaciones tales como: Motorola, Samsung, HTC, LG, Sony Ericsson… Pero Android es también un lenguaje de programación y un framework para desarrollo de aplicaciones.

Es decir, que al conjunto de todos esos elementos se lo llama Android. Con Android colaboran muchísimas personas alrededor del mundo debido a que es un proyecto de Software Libre.

La sintaxis del lenguaje de programación es Java, pero es importante destacar (y marcar la diferencia) que **no es Java**, ya que no implementa todas las bibliotecas de ME.

Así que mostremos lo diferente o igual que puede ser el desarrollo de esta aplicación Android.

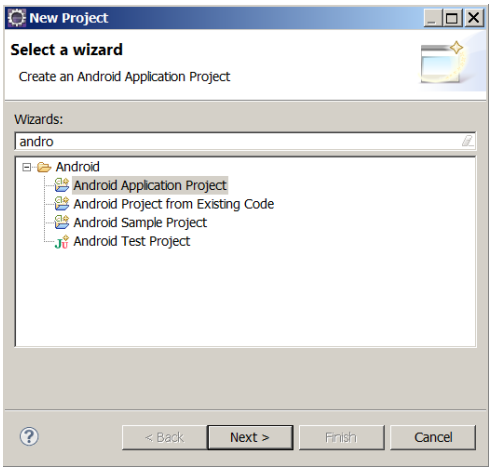
Creación proyecto

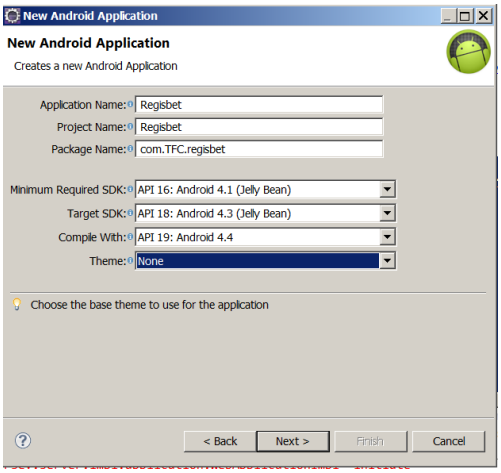
Para crear este proyecto hemos tenido que realizar los siguientes pasos.

File->New->Project->Se escoge “Android Application Project”

Aparecerá otra pantalla indicando el nombre de la aplicación, el nombre del proyecto y el package por defecto.

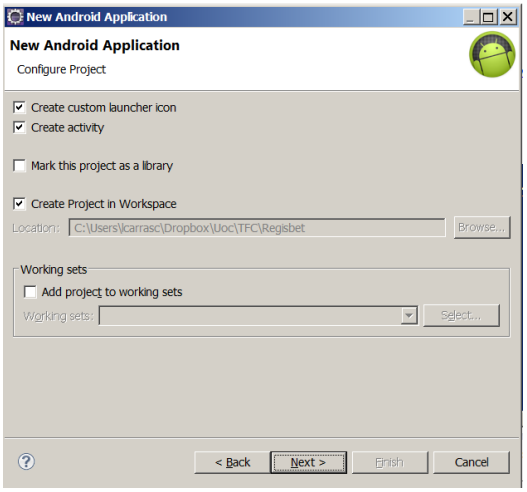
El apartado más importante es el de mínimo de versión de SDK. Ello indica que soporte va a llegar a tener la aplicación dentro de toda la gama de versiones existentes. En nuestro caso se escoge la API 16: Android 4.1 (Jelly Bean). Se clica sobre Next.

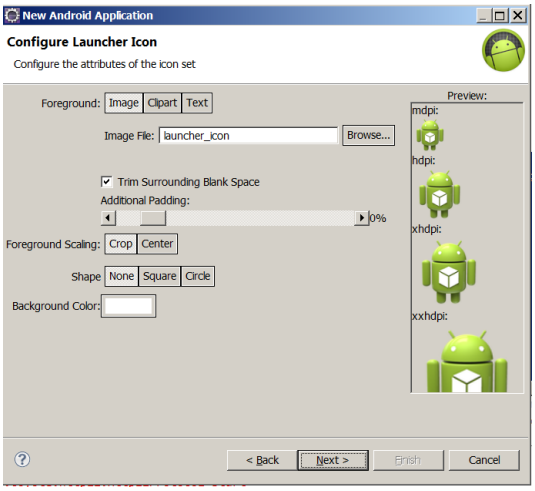




Se muestra la siguiente pantalla de configuración para que tenga un icono, una actividad de principio e indicas donde quieres que se guarde.

Cuando clicas sobre Next se muestra la pantalla de configuración de Icono.

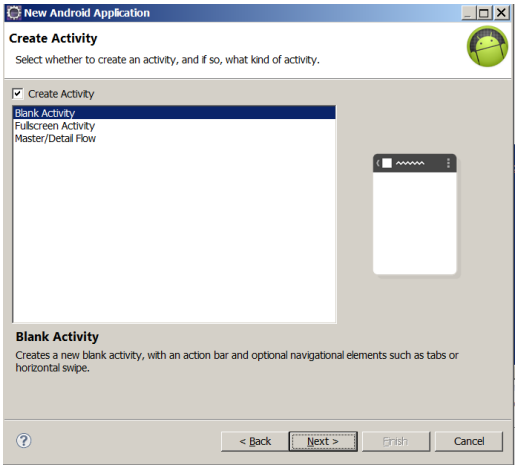


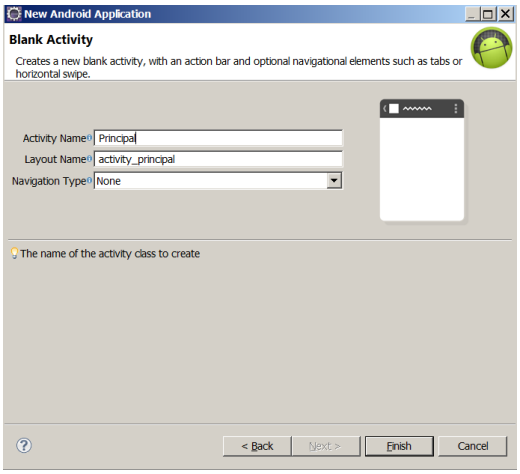


Se clica sobre Next y aparece las pantallas de configuración de la activity que anteriormente habíamos clicado que se crease.

En nuestro caso al ser noveles y viendo que se crean activities con un tipo de diseño bastante complicadas por este wizard, se decide que sea sin tipo para facilitar el entendimiento inicial.

Cuando se clica Next sobre esta pantalla se muestra el nombre deseado para la activity, el nombre del layout y el tipo de navegación, en nuestro caso None, para facilitar establecer un conocimiento inicial más rápido y sencillo.





En su momento me costó entender los conceptos básicos que cualquier tutorial de inicio te marca, así que con un poco de vergüenza y orgullo voy a poner los conceptos que para mí hoy son básicos pero que en su día me costó de entender.

- **Activity**: Las aplicaciones que tengan interfaces gráficas deberán tener al menos una clase del tipo Activity, ya que ésta actúa como lo que comúnmente se conoce como "formulario". En una Activity se colocan los elementos de la interfaz gráfica.

- **Intents**: Es un mecanismo para comunicar a las distintas aplicaciones y Activities. Android esta desarrollado sobre la base de reutilizar código y aplicaciones existentes, es por eso que esta característica es tan importante.
- **Content Providers**: Es el mecanismo encargado de administrar la información que se pretende que perdure.
- **Broadcast Receivers**: Se utilizan para que una aplicación responda a un determinado evento del sistema. En nuestra aplicación no lo utilizamos.
- **Services**: Son lo que comúnmente se conocen como procesos. Estos seguirán corriendo aunque no haya una interfaz gráfica para mostrar la aplicación. De momento no implementado en la app.

Una vez establecidos los nuevos conceptos y tras varios proyectos básicos de prueba ya podía empezar a trastear y generar algo de código útil.

Implementación del Código

El principio de toda aplicación Android parte del fichero AndroidManifest.xml de aquí se otorgan todos los privilegios que puede tener la aplicación.

En nuestro caso como tenemos interacción con la cámara de fotos, con la agenda y envío de datos mediante conexión a internet debemos de establecer estos permisos.

Para ello debemos editar dicho AndroidManifest.xml e insertar lo siguiente:

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
<uses-permission android:name="android.permission.INTERNET"/>
```

Con estas cuatro líneas tenemos privilegios para poder leer los contactos de la agenda, acceder a la cámara y tener conexión a internet.

Como hemos realizado en la anterior configuración, nuestra primera clase que se ejecute será principal.java que llamara a la activity activity_principal.

Esta pantalla mostrará un listado de las apuestas que están vigentes.

Al iniciarse la actividad se inicia el ciclo de vida de creación. Es muy importante tener claro el concepto de ciclo de vida.

Todas las actividades siempre se inician con el método onCreate, pero se utilizará el ciclo de vida de una actividad para ejecutar unas instrucciones u otras según se requiera en cada estado de la activity.

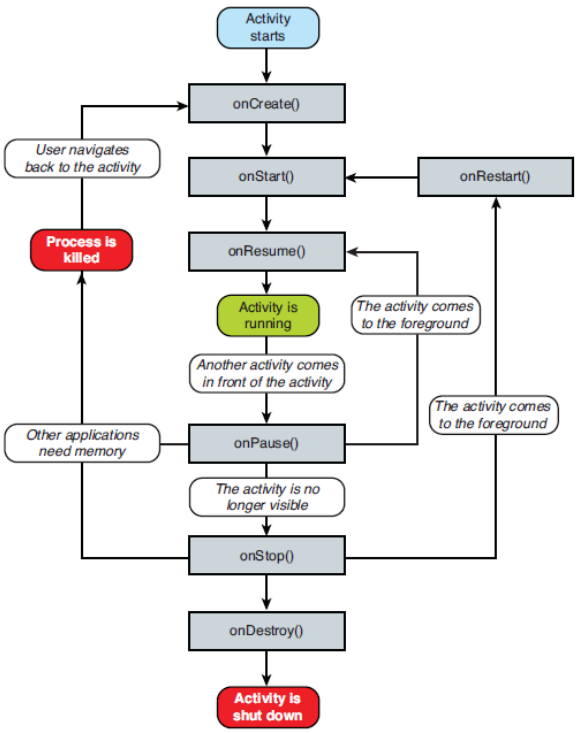


Figure 2.2 Activity Lifecycle from http://developer.android.com/.

La clase principal.java en el método onCreate realizará las siguientes funciones:

- Establece una tarea periódica cada minuto para que vaya refrescando la lista de apuestas.
- Recibe todos los datos de las apuestas mediante una llamada asíncrona a la clase `ApuestasList`.
- Establece que se debe llamar a la clase de BetStateActivity cuando una apuesta ha sido seleccionada para ver o modificar su contenido.
- Llama al método `getOverflowMenu` que realiza la tarea de mostrar los tres puntos del menú para que se muestren más opciones. En nuestro caso no se utiliza.
- Llama al método `manageUser` que realiza la creación del usuario y la recogida del número de teléfono para tener el control del usuario. Se guarda en el sharedPreferences que es una BBDD clave valor de gestión interna que proporciona Android para este tipo de tareas.

```
SharedPreferences settings = getSharedPreferences(SHARED_PREF, Context.MODE_PRIVATE);
        idUsuario = settings.getString(getString(R.string.idUsuario), null);
        if (idUsuario == null)     {

                //En caso que no exista se crea
                SharedPreferences.Editor editor = settings.edit();

                double random = Math.random();
                String id = String.valueOf(random).substring(String.valueOf(random).length()-
10,String.valueOf(random).length());

                editor.putString(getString(R.string.idUsuario), id);

                //Insertamos tambien el número de teléfono para controlar de otra forma el usuario.
                TelephonyManager tm = (TelephonyManager)getSystemService(TELEPHONY_SERVICE);
                String number = tm.getLine1Number();
                editor.putString(getString(R.string.phoneNumber), id);
                editor.commit();
                idUsuario = id;
        }
```

Principal.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_principal);

        con = this;
        act = this;

        // S - Realizar tarea en background
        HandlerThread hThread = new HandlerThread("HandlerThread");
        hThread.start();
        Looper loop = hThread.getLooper();
        final Handler handler = new Handler(loop);
        final long oneMinuteMs = 60 * 1000 ;

        Runnable eachMinute = new Runnable() {
                @Override
                public void run() {
                        Log.d("Runnable", "Each minute task executing");

                        ApuestasList task = new ApuestasList(act);
                    task.execute(new String[] { "http://192.168.1.128:8080/WebService/rest/ApuestaList"
});

                        handler.postDelayed(this, oneMinuteMs);
                }
        };
        // E - Realizar tarea en background

        // Llamada al background por primera vez
        handler.postDelayed(eachMinute, oneMinuteMs);

        lista = (ListView) findViewById(R.id.listaapuestas);

        ApuestasList task = new ApuestasList(this);
    task.execute(new String[] { "http://192.168.1.128:8080/WebService/rest/ApuestaList" });


        //Miramos que Apuesta ha escogido.
        lista.setOnItemClickListener ( new AdapterView.OnItemClickListener ( )
        {
                @Override
            public void onItemClick ( AdapterView <?> aView, View v, int position, long id )
            {

                        AMainList bet = (AMainList)aView.getItemAtPosition(position);

                        Intent intent = new Intent(con, BetStateActivity.class);
                        intent.putExtra(BET_CHOSED, bet.getApuestaId());
                        startActivityForResult(intent, BET_STATE_ACTIVITY);

            }
        });

        getOverflowMenu();

        manageUser();

}
```

Como se ha comentado antes existe una línea que hace referencia a que activity se va a cargar `setContentView(R.layout.activity_principal);` si accedemos a los resources hace referencia al activity llamado activity_principal.xml.

Su contenido es el siguiente:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".Principal" >

    <TextView
        android:id="@+id/textointroductoriolista"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="@dimen/paddingRL"
        android:text="Listado de Apuestas Vigentes"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <ListView
        android:id="@+id/listaapuestas"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        tools:listitem="@layout/itemlista" >
    </ListView>

</LinearLayout>
```

Se puede observar que dentro del linearlayout hay dos componentes.

El más importante es el que pone Listview ya que es el que indica que es un Listview llamado ítemlista y que contendrá los valores de la apuesta. Los datos que se muestran son:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="@dimen/paddingRL" >

    <ImageView
        android:id="@+id/fotoListPrincipal"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:adjustViewBounds="true"
        android:cropToPadding="true"
        android:scaleType="center" />

    <TextView
        android:id="@+id/tituloApuestaList"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/descripcionApuestaList"
        android:layout_marginBottom="5dp"
        android:layout_marginLeft="3dp"
        android:layout_toRightOf="@+id/fotoListPrincipal"
        android:text="Large Text"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/descripcionApuestaList"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/fotoListPrincipal"
        android:layout_alignLeft="@+id/tituloApuestaList"
        android:layout_alignRight="@+id/tituloApuestaList"
        android:layout_marginBottom="3dp"
        android:text="Medium Text"
        android:textAppearance="?android:attr/textAppearanceSmall" />

</RelativeLayout>
```

Este Relative layout cuenta con una imagen a la izquierda con un pequeño margen y dos textviews a su lado. Siendo algo parecido a esta imagen:

La clase principal.java realiza la llamada de forma asíncrona para recoger estos valores, pero en alguna ocasión las imágenes no se mostraban y era por que se mostraba toda la pantalla antes de haber recibido las imágenes.

Esto se solucionó con el siguiente código:

```
@Override
            protected String doInBackground(String... urls) {
                 String url = "http://192.168.1.128:8080/WebService/rest/ApuestaList";

                 String tag = "REST Request Select";

                          try {
                                   //Realizamos la petición Get para recibir todas las apuestas
                                   Log.d(tag, "Request: " + url);
                                   final HttpParams httpParams = new BasicHttpParams();
                                   HttpClient httpClient = new DefaultHttpClient(httpParams);
                                   HttpGet get = new HttpGet(url);
                                   get.setHeader("content-type", MediaType.APPLICATION_JSON);

                                   HttpResponse resp = httpClient.execute(get);
                                   String respStr = EntityUtils.toString(resp.getEntity());
                                   Log.d(tag, "Response from get: " + respStr);
                                   return respStr;

                          } catch (Exception ex) {
                                   Log.e(tag, "Error: " + ex.getMessage());
                          }
                          catch (OutOfMemoryError out) {
                                   Log.e(tag, "consultaString OutOfMemoryError: " + out.getMessage());
                          }
                          return null;
            }
```

Realizamos la llamada al webservice mediante @GET.

En `onPostExecute` recibimos los datos que nos ha devuelto la llamada al Webservice y lo tratamos llamando a otra clase de forma asíncrona para que se puedan recibir las imágenes del servidor.

Con la llamada se le pasa por parámetros toda la lista de apuestas que se van a mostrar.

```
    @Override
    protected void onPostExecute(String response) {

         // Tratamos respuesta, pasamos a JSON y mapeamos a la clase correspondiente
         ArrayList<AMainList> mainApuestaList = new ArrayList<AMainList>();
                            JSONObject json = null;
                            AMainList[] apuestaList = null;
                   try {
                    // Comprobamos que no haya dado error
                   if (response != null) {
                            json = new JSONObject(response);

                    String amainList = (String) json.get("response");
                    ObjectMapper mapper = new ObjectMapper();
                    apuestaList = mapper.readValue(amainList, AMainList[].class);

                    for (AMainList apuesta : apuestaList)
                                      mainApuestaList.add(apuesta);
          //cuando tenemos todas las apuestas cargadas en memoria, Cargamos las fotos.
          //Cuando se hayan cargado todas las fotos se cargará el listView (onPostExecute)
                    if (mainApuestaList.size() > 0) {
                              ImagenServerAsync task = new ImagenServerAsync (mainApuestaList, parent);
                              task.execute(new String[] { });
                    }

         } else {
                    Log.e("ListView", "Error getting apuestas: " + json.getJSONObject("object"));
          }

         } catch (IOException e) {
                   // TODO Auto-generated catch block
                   e.printStackTrace();
         }
    }
}
```

La clase llamada asíncronamente recibe los parámetros los recorre y por cada una de las apuestas realiza una petición al servidor Web de la imagen, en nombre se encuentra en la lista de apuestas recibidas por parámetro.

```
@Override
    protected String doInBackground(String... urls) {

         String tag = "REST Request Image";

         try {

                   for (AMainList apuesta : mainApuestaList){
                            String fotoName = apuesta.getFoto();
                                      if (fotoName != null) {
                                      apuesta.setFotoD(Image.recoverImageServer(fotoName + ".jpg"));
                                      }
                   }
                   } catch (Exception ex) {
                            Log.e(tag, "Error: " + ex.getMessage());
                   }
                   catch (OutOfMemoryError out) {
                            Log.e(tag, "consultaString OutOfMemoryError: " + out.getMessage());
                   }
                   return null;

    }

    @Override
    protected void onPostExecute(String response) {

         //cuando se han recuperado todas las fotos del ListView es cuando se lanza para que no se quede
         //ninguna foto sin refrescar.
                   // Creo el adapter personalizado
                   AMainListAdapter adapter = new AMainListAdapter(this.parent, mainApuestaList);

                   // Lo aplico
                   lista.setAdapter(adapter);

    }
}
```


Hasta que no ha recuperado todas las imágenes es cuando hace la llamada al adapter para que así en su getView lo cargue todo.

En el método onCreate de la clase Principal.java se establece que cuando se clique sobre alguna apuesta se muestre el detalle y dependiendo de si eres el creador o no te permite modificar los datos de la apuesta.

La clase que muestra los datos de la apuesta es `BetStateActivity`.

BetStateActivity.java

Esta activity muestra los datos de la apuesta.

Si accede el creador permitirá modificar los campos, pero sino es el creador todos los campos aparecerán protegidos y únicamente podrá comprobar cual es el estado de la apuesta.

Para mostrar los datos lo primero que hace la clase BetStateActivity.java es iniciar una llamada asíncrona a la clase `ApuestasChange` desde el método onCreate.

El método realiza la llamada de forma asíncrona al WebService clase

```
@Override
    protected String doInBackground(String... urls) {
         String url = urls[0];

                  String tag = "REST Request Select";

                  try {

                           Log.d(tag, "Request: " + url);
                           final HttpParams httpParams = new BasicHttpParams();
                           HttpConnectionParams.setConnectionTimeout(httpParams, 25000);
                           HttpConnectionParams.setSoTimeout(httpParams, 25000);
                           HttpClient httpClient = new DefaultHttpClient(httpParams);

                           //Dependiendo de cuantos parámetros tenga se hace una llamada GET o PosT
                           //1 param se hace llamada GET para recibir la apuesta existente
                           //2 params llamada POST para realizar el update.
                           if (urls.length == 1) {
                                    HttpGet get = new HttpGet(url + "/" + betId);
                                    get.setHeader("content-type", MediaType.APPLICATION_JSON);
                                    try {

                                             HttpResponse resp = httpClient.execute(get);
                                             String respStr = EntityUtils.toString(resp.getEntity());
                                             Log.d(tag, "Response from get: " + respStr);
                                             return respStr;
                                    } catch (Exception ex) {
                                             Log.e(tag, "Error: " + ex.getMessage());
                                    }
                           }
                           else if (urls.length == 2){
                                    MainApuestaREST mAR = new MainApuestaREST();
                           try {
                                             return mAR.postRequest("MainApuestaU", apuestaString);
                                    } catch (Exception e) {
                                             // TODO Auto-generated catch block
                                             e.printStackTrace();
                                    }
                           return null;
                           }
                  } catch (OutOfMemoryError out) {
                           Log.e(tag, "consultaString OutOfMemoryError: " + out.getMessage());
                  }

                  return null;
    }
```

Realiza las llamadas de select o Update dependiendo de los parámetros que se le pase. Viéndolo así se podría haber realizado la creación de otra clase para la diferenciación de la llamada a los Web Services pero de esta forma queda todo más compacto.

Cuando recibe un parámetro realiza el select para recuperar los datos de la BBDD. Cuando recibe dos parámetros realiza el Update.

Para el control de como han ido los procesos si todo haya ido correctamente el retorno del select se devuelve en el objeto OFObjectResponse y se recoge con `json = new JSONObject (response).equals ("0")` en este caso es 0 y quiere decir que es correcta la respuesta, en el caso incorrecto se devuelve -1. Si el Update ha ido correctamente obtendremos un retorno de 1 y en caso que haya ido mal un -1.

El tratamiento que recibimos en el código es el siguiente:

```
@Override
protected void onPostExecute(String response) {

        // Tratamos respuesta, pasamos a JSON y mapeamos a la clase correspondiente
        JSONObject json = null;

        try {

         // Comprobamos que no haya dado error
        if (response != null) {
                json = new JSONObject(response);
                //cuando devuelve -1 es cuando hay error
                //En este caso es porque el usuario existe y se recuperan los datos de la BBDD.
                if (json.getString("error").equals("0")){
                        setApuestaToActivity(response);
                }
                else if (json.getString("error").equals("1")){
                Toast.makeText(con, "Apuesta modificada correctamente", Toast.LENGTH_SHORT).show();
                finish();
                } else {
                         Log.e("BetState", "Error getting apuesta: " + json.getJSONObject("object"));
                         Toast.makeText(con, R.string.errorGettingApuesta, Toast.LENGTH_SHORT).show();
                 }
        }
```

Si el apartado error del objeto OFObjectResponse devuelto es 0 parseamos lo devuelto en la función `setApuestaToActivity(response);` donde encuentra todos los componentes de la pantalla y los carga con los datos recibidos.

En caso que hayamos recibido un 1 en el apartado error del objeto OFObjectResponse quiere decir que el Update se ha realizado de forma correcta.

En cualquier otro caso quiere decir que ha habido algún problema.

Acciones de BetStateActivity y BetNewActivity

BetNewActivity se encarga de insertar los datos principales de la apuesta para introducirlos en la BBDD.

BetStateActivity se encarga de recuperar los datos de la apuesta y en caso de tener privilegios modifica los datos.

La diferencia entre las dos clases es que BetState recoge al iniciar los datos de BBDD y la activity de nueva apuesta no. También varía un poco el tratamiento de la imagen, la inserción de la fecha y la edición de las respuestas.

Son cambios importantes en el funcionamiento de la aplicación ya que sino mal funcionaría, pero son cambios que tecnológicamente no son muy importantes, es por ello que trataremos estas dos pantallas hablando únicamente de BetStateActivity que es más compleja que BetNewActivity.

En el caso que sea el creador de la apuesta podrá cambiar cualquier cosa de la apuesta.

Titulo y descripción de la apuesta aparecen habilitados para la modificación igual que el resto de campos.

En el caso que no sea el creador aparecerán todos los campos deshabilitados y un mensaje indicando que no podrá realizar ningún cambio por que no es el creador de la apuesta.

En el caso que se nos permita cambiar los datos hay diversos eventos a tener en consideración.

Apuesta Contra

Cuando se clica sobre el botón de añadir adversario, se abre una pantalla con todos los contactos de la agenda, a medida que se van seleccionando se van añadiendo adversarios hasta un máximo de 10.

La llamada a esta activity espera de la respuesta para cargar los datos en el campo contra quien se realizará la apuesta.

```
public void addContacts(View view) {

        Intent intent = new Intent(this, ContactsListActivity.class);
        startActivityForResult(intent, RETURN_VERSUS);
}
```

Cuando devuelve los resultados se tratan en el método onActivityResult que se llama cuando se sale de la activity pertinente.

Más adelante lo veremos con más atención.

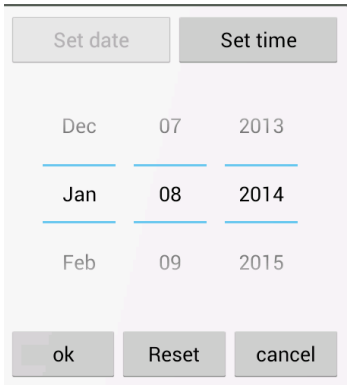
Fecha Finalización

Cuando se clica sobre el botón de Fecha se muestra un dialog donde aparece la fecha y el tiempo, cuando se marque se estará estableciendo la finalización de la apuesta.

```
public void showDateTimeDialog(View view) {
        final Dialog localDialog = new Dialog(this);
        RelativeLayout localRelativeLayout = (RelativeLayout) getLayoutInflater()
                        .inflate(R.layout.date_time_dialog, null);
        final DateTimePicker localDateTimePicker = (DateTimePicker) localRelativeLayout
                        .findViewById(R.id.DateTimePicker);
        //Para que se muestre la hora en 24 horas true para que se muestre en formato PM AM false
        boolean bool = true;
        //Controlamos en el dateTimePicker que la fecha es la actual más 5 minutos.
        Button bb = (Button) localRelativeLayout.findViewById(R.id.SetDateTime);
        bb.setOnClickListener(new View.OnClickListener() {
                public void onClick(View paramAnonymousView) {
                        String fechaInsertada = localDateTimePicker.get(5) + "/"
                                        + (localDateTimePicker.get(2) + 1) + "/"
                                        + localDateTimePicker.get(1);

                        TextView date = ((TextView) findViewById(R.id.Date));
                        if (localDateTimePicker.is24HourView()) {
                                StringBuilder localStringBuilder = new
StringBuilder(String.valueOf(localDateTimePicker.get(11))).append(":")
                                                .append(localDateTimePicker.get(12));
        //Comprobamos que la fecha introducida es mayor a la actual más 5 minutos (pasados por parametro)
        //teniendo en cuenta los segundos por eso es 4.
                        int mayor = controlDateTimeInMinutes(fechaInsertada, localStringBuilder, 4);
                        if (mayor == 1) {
                                date.setText(fechaInsertada);
                        ((TextView) findViewById(R.id.Time)).setText(localStringBuilder.toString());
                                localDialog.dismiss();
                                        return;
                                }
                                else
                        Toast.makeText(con, R.string.errorFechaFinApuesta, Toast.LENGTH_SHORT).show();
                        }

                        TextView localTextView = (TextView) findViewById(R.id.Time);
                        StringBuilder localStringBuilder = new StringBuilder(String
                                        .valueOf(localDateTimePicker.get(10))).append(":")
                                        .append(localDateTimePicker.get(12)).append(" ");

                        //Este apartado no debe aplicar se muestra en formato 24H
                        if (localDateTimePicker.get(9) == 0)
                                ;
                        for (String str = "AM";; str = "PM") {
                                localTextView.setText(localStringBuilder.append(" ") + str);
                                break;
                        }
        }
```


```
public void showDateTimeDialog(View view) {
/**** Inserta los minutos pasados por parámetros y comprueba que la fechaTime sea mayor que la fecha           *
@param fechaInsertada       * @param localStringBuilder * @param minutes

* @return -1 si la fecha introducida es menor a la acutal, 0 i es igual y 1 si es mayor               */

                    private int controlDateTimeInMinutes(String fechaInsertada, StringBuilder localStringBuilder, int
minutes) {
                              //Controlamos en el dateTimePicker que la fecha es la actual más 5 minutos.
                              Date dateInserted = null;
                              Date fechaActualMas5Min = new Date();
                              //Insertamos 5 minutos más.
                              long t= fechaActualMas5Min.getTime();
                              fechaActualMas5Min =new Date(t + (minutes * ONE_MINUTE_IN_MILLIS));

                              String timeString = localStringBuilder +":00";
                              SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

                              try {
                                        dateInserted = formatter.parse(fechaInsertada + " " + timeString);

                              } catch (ParseException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                              }
                              return dateInserted.compareTo(fechaActualMas5Min);
                    }
          });                 SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

                                        try {
                                        dateInserted = formatter.parse(fechaInsertada + " " + timeString);

                                                  } catch (ParseException e) {
                                                           // TODO Auto-generated catch block
                                                           e.printStackTrace();
                                                  }
                                                  return dateInserted.compareTo(fechaActualMas5Min);
                                        }
                              });
```

A pesar de que el código queda un poco confuso en este documento, en el IDE Eclipse no lo es tanto en este caso he tenido que editarlo para que quepa todo en la misma pantalla y así quede como en un mismo bloque.



Aun así creo que se puede entender que es la lógica para el tratamiento y formateo de las fecha/hora, se puede configurar en que formato se quiere y que hace cuando se clica sobre los diferentes botones que aparecen por pantalla.

Cuando se clica sobre Set date o Set time se muestra la fecha o la hora que se cargó la activity.

Al apretar sobre reset establece la hora actual.

Cuando clicas sobre cancel hace desaparecer el dialog sin ningún tipo de modificación en la pantalla padre.

En el momento que se clica el botón ok es cuando se ejecuta el siguiente código.

```
//Comprobamos que la fecha introducida es mayor a la actual más 5 minutos (pasados por parametro)
        //teniendo en cuenta los segundos por eso es 4.
        int mayor = controlDateTimeInMinutes(fechaInsertada, localStringBuilder, 4);

        if (mayor == 1) {
                date.setText(fechaInsertada);
                ((TextView) findViewById(R.id.Time)).setText(localStringBuilder.toString());
                localDialog.dismiss();
                return;
        }
        else
                Toast.makeText(con, R.string.errorFechaFinApuesta, Toast.LENGTH_SHORT).show();
```

Comprueba que la fecha sea 5 minutos posterior a la fecha actual. Es un parámetro establecido y que se puede modificar en caso de ser necesario.

En caso que el tiempo introducido sea correcto se guardan los datos en la pantalla principal y se desaparece el dialog.

Inserción de Imagen

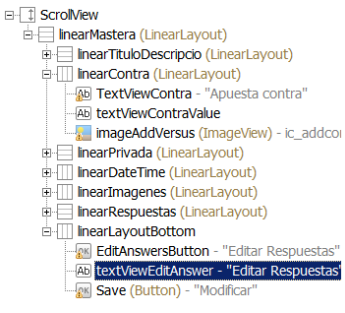
Cuando se quiere realizar una inserción de una imagen se clica sobre el link azul este mostrará un Alertdialog indicando cual es la fuente de donde se quiere recoger o general la imagen.

```
public void captureImageGallery(View view) {

        AlertDialog.Builder builder = new AlertDialog.Builder(con);
        builder.setTitle("Choose Image Source");
builder.setItems(new CharSequence[] { "Gallery", "Camera" },
                new DialogInterface.OnClickListener() {

        @Override
        public void onClick(DialogInterface dialog, int which) {
                switch (which) {
                case 0:

                        Intent pictureActionIntent = new Intent(
                                        Intent.ACTION_GET_CONTENT, null);
                        pictureActionIntent.setType("image/*");
                        pictureActionIntent.putExtra("return-data", true);
                        startActivityForResult(pictureActionIntent,
                                        CAPTURE_IMAGEGALLERY_ACTIVITY_REQUEST_CODE);

                        break;

                case 1:
                        Intent intentCamera = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                        fileUri = Image.getOutputMediaFileUri(Image.MEDIA_TYPE_IMAGE);
                        // create a file to save the image
                        intentCamera.putExtra(MediaStore.EXTRA_OUTPUT,
                                        fileUri);
                        // set the image file name
                        // start the image capture Intent
                                startActivityForResult(intentCamera,
                                                CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE);

                                break;

                        default:
                                break;
                        }
                }
        });
```

Para mi lo más tedioso fue como recoger la imagen, esto se realiza en el apartado del onActivityResult que más adelante trataremos.
Se debe diferenciar cual es la procedencia de la imagen si es de cámara o si es de la tarjeta sd.
En ambos casos lo que realmente varia es la fuente de procedencia pero se le tiene que dar trato diferenciativo.

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

        if (requestCode == RESULTS && resultCode == RESULT_OK) {
                setOptionsFromEditActivity(data);
        } else if (requestCode == CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE) {
                if (resultCode == RESULT_OK) {
                        // Image captured and saved to fileUri specified in the Intent
                Toast.makeText(this, "Image saved to:\n" + data.getData(),
        Toast.LENGTH_LONG).show();
                setImageFromMntSDCard(data);
        } else if (resultCode == RESULT_CANCELED) {
                // User cancelled the image capture
        } else {
                // Image capture failed, advise user
        }
} else if (requestCode == CAPTURE_VIDEO_ACTIVITY_REQUEST_CODE) {
        if (resultCode == RESULT_OK) {
                // Video captured and saved to fileUri specified in the Intent
                Toast.makeText(this, "Video saved to:\n" + data.getData(),
        Toast.LENGTH_LONG).show();
        } else if (resultCode == RESULT_CANCELED) {
                // User cancelled the video capture
        } else {
                // Video capture failed, advise user
        }
} else if (requestCode == CAPTURE_IMAGEGALLERY_ACTIVITY_REQUEST_CODE) {
        Toast.makeText(this, "Image recovered from:\n" + data.getData(), Toast.LENGTH_LONG).show();
        setImage(data);

}
```

Cuando el resultado viene de la acción de la cámara (`CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE`) o de la galería de imágenes (`CAPTURE_IMAGEGALLERY_ACTIVITY_REQUEST_CODE`) se inserta la imagen en la imageView de la activity y se actualiza la variable `pathImage` de la activity indicando cual es el nuevo Path de la imagen. Está variable se utilizará para realizar la conversión de la imagen para su posterior guardado en el servidor.



La gran limitación visual que tiene esta activity es que por falta de experiencia y destreza a la hora de realizar un diseño válido para todas las plataformas, realicé un diseño que se adecuaba a la resolución de pantalla del móvil y la máquina virtual de Android.

Para ello implementé un scrollView para que si hay alguna resolución más baja se pudiese con el scroll, pero el resto de la pantalla es totalmente estática y realizada con linearLayouts, tanto horizontales como verticales.

Por lo que este es un gran punto de mejora.

Editar Respuestas

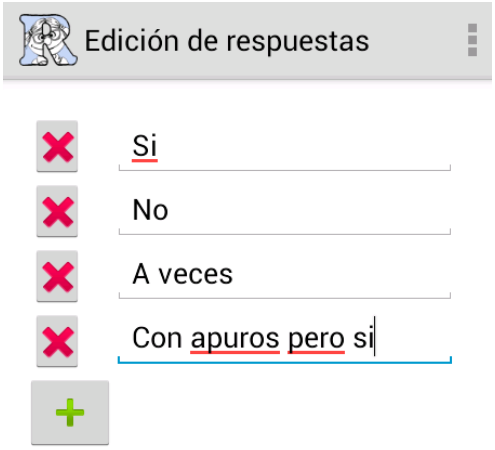
Cuando se clica sobre esta acción se muestra la activity Edit_AnswerActivity.

Este lo que hace es recorrer todas las respuestas de la pantalla BetStateActivity y las envía a la activity de edición.

```java
/***
 * Pone en el intent las respuestas posibles y las envia a la pantalla de
 * edición de respuestas.
 *
 * @param view La vista.
 */
public void editAnswers(View view) {
        Intent intent = new Intent(this, EditAnswer_Activity.class);

        TextView textAnswer1 = (TextView) findViewById(R.id.textRespuesta1);
        TextView textAnswer2 = (TextView) findViewById(R.id.textRespuesta2);

        if (textAnswer1 != null) {
                respuestaValues = new TreeMap<String, String>();
                intent.putExtra(0 + "", textAnswer1.getText().toString());
                respuestaValues.put("1", textAnswer1.getText().toString());

        }
        if (textAnswer2 != null) {
                intent.putExtra(1 + "", textAnswer2.getText().toString());
                respuestaValues.put("2", textAnswer1.getText().toString());
        }

        int parametros = 2;
        TextView textView = (TextView) findViewById(parametros);
        while (textView != null) {
                intent.putExtra(parametros + "", textView.getText().toString());
                respuestaValues.put((parametros + 1) + "", textAnswer1.getText()
                                .toString());
                parametros++;
                textView = (TextView) findViewById(parametros);

        }

        startActivityForResult(intent, RESULTS);

}
```

Establecemos todos los valores de las respuestas con putExtras (clave / valor) para pasar la información a la activity de Edición de respuestas.

EditAnswer_Activity.java



En esta pantalla se muestran las posibles respuestas que puede tener una apuesta. Se puede insertar hasta 10 opciones. Más se entiende como que no es necesario, son demasiadas opciones y puede confundir al usuario.

Para crear hasta las 10 opciones todos los compentes de las respuestas se crean dinámicamente.

A continuación se muestra el código que crea los componentes que se reciben de la pantalla betStateActivity o betNewActivity.

```
//Recogemos las opciones de la pantalla padre (BetState or BetNewActivity)
Bundle bundle = getIntent().getExtras();
RadioButton button;
EditText edit;
if (bundle != null) {
         //Los insertamos
         for ( i = 0; bundle.getString(i + "") != null ; i++) {

         //Insertamos dentro del vector para controlar los valores
         String contenido = bundle.getString(i + "").substring(3, bundle.getString(i + "").length());
                  radioValues.put( i + "", contenido);

                  //Insertamos botón de borrar
                  ImageButton ib = new ImageButton(this);
             ib.setId(i);
             Drawable ic_delete = getResources().getDrawable(android.R.drawable.ic_delete);
             ib.setImageDrawable(ic_delete);
             LayoutParams params = new LayoutParams(50, 60);
             ib.setLayoutParams(params);

             ib.setOnClickListener(new OnClickListener() {
                  //Controlamos la acción para los botones creados al cargar la pantalla.
                 public void onClick(View v) {
                  ImageButton ib = (ImageButton) findViewById(v.getId());
                  Toast.makeText(con, "Opción Borrada" , Toast.LENGTH_SHORT).show();
                  linearDelete.removeView(ib);

                  EditText et = (EditText) findViewById(v.getId()+100);
                  linearEditText.removeView(et);

                  radioValues.remove(v.getId()+"");

                 }
             });

             linearDelete.addView(ib);

             // Creamos el EditText

             EditText et = new EditText(this);
             et.setId(i+100);
             et.setText(contenido);

             linearEditText.addView(et);

         }
}
```

Con el código `ImageButton ib = new ImageButton(this);` y `EditText et = new EditText(this);` se crean los components y con el código `linearEditText.addView(et);` se inserta por código el componente creado dentro del layout.

Cuando se crea el botón de borrado se debe de establecer una acción que borre los componentes creados dinámicamente.

Con la sentencia `ib.setOnClickListener(new OnClickListener() {` establecemos que cada vez que se clica el botón para eliminar la respuesta se ejecute el código de borrado `radioValues.remove(v.getId()+"");`. Al clicar sobre el botón de guardar se comprueba que tenga como mínimo una respuesta, sino se muestra error, en caso correcto se recorre la variable `radioValues` que contiene todas las opciones de la pantalla se inserta en la variable intent `resultData.putExtra(contadorInterno + "", etValue);` y si todo ha ido correctamente se cierra `finish();`

En la activity de Nueva apuesta o edición de apuesta recuperará mediante onActivityResult las opciones y las cargará en pantalla.

```
if (radioValues.size() > 0){
        Intent resultData = new Intent();
        boolean todosVacios = true;
        //Si los botones de edición no están habilitados quiere decir que se deben guardar los cambios.
        //Miramos cual es el último valor de las respuestas y en base a eso hacemos el recorrido
        Entry<String, String> lastEntry = radioValues.lastEntry();
        int maxRespuestas = Integer.valueOf(lastEntry.getKey());
        int contadorInterno = 0;
        for (int i = 0 ; maxRespuestas >= i ; i++) {

                EditText et = (EditText) findViewById(i + 100);
                if (et != null) {
                        String etValue = et.getText().toString();
                        //Si el campo no está vacio se inserta dentro del listado
                        if (!etValue.trim().isEmpty()){

                                resultData.putExtra(contadorInterno + "", etValue);
                                todosVacios = false;
                                contadorInterno++;
                        }
                }

        }
        if (!todosVacios) {
                setResult(Activity.RESULT_OK, resultData);
                finish();
        }
}
else {
        Toast.makeText(con, R.string.masRespuestas, Toast.LENGTH_SHORT).show();
}
```

BetPersonalData.java

Es la pantalla que donde se indican cuales son los datos de la persona dentro de la aplicación.

Los campos se muestran son una imagen, el nick y la descripción.

La primera vez que se entra en esta pantalla, como no hay datos de usuario se muestran todos los componentes vacíos.

Pero si se realiza un guardado la siguiente vez que se entra se recuperan los datos y se permite la edición.

Como peculiaridad con respecto a las otras clases es que esta las tres llamadas a los Web Services el select, el insert y el Update.

Para ello el tratamiento que se hace es una llamada asíncrona a la clase con diferenciación de parámetros.

```
// Llamada a WS REST
try {
        ObjectMapper mapper = new ObjectMapper();
        personaString =  mapper.writeValueAsString(persona);

        Button ib = (Button) findViewById(R.id.buttonGuardarPersona);

        //Si el botón que se muestra es guardar se debe realizar una inserción en la tabla del usuario.
//Si el botón muestra modificar se debe realizar una modificación.
if (ib.getText().toString().equals(getString(R.string.guardar))){

        Persona task = new Persona(this);
        //Como no funciona el modo Post (a saber porque). Se utiliza el modo asíncrono.
    task.execute(new String[] { "http://192.168.1.129:8080/WebService/rest/Persona", "" });

}
else if (ib.getText().toString().equals(getString(R.string.modificar))){
        Persona task = new Persona(this);
        //Como no funciona el modo Post (a saber porque). Se utiliza el modo asíncrono.
    task.execute(new String[] { "http://192.168.1.129:8080/WebService/rest", "", "" });
}
```

La sentencia select se ejecuta en el onCreate de la clase ya que recupera los datos.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_personal_data);

        // Restore preferences para recoger el idUsuario
        SharedPreferences settings = getSharedPreferences(Principal.SHARED_PREF, Context.MODE_PRIVATE);
        idUsuario = settings.getString(getString(R.string.idUsuario), null);
        phoneNumber = settings.getString(getString(R.string.phoneNumber),null);

        con = this;

        Persona task = new Persona(this);
    task.execute(new String[] { "http://192.168.1.129:8080/WebService/rest/Persona" });

}
```

Recupera el usuario y el teléfono para poder realizar la sentencia de select.

El código de la inserción de las imágenes tanto la thumbail que se muestra en el imageView como la que se muestra cuando se clica es la el siguiente:

```
//S - Sección imagen
if (pathImage != null) {
        //Recogemos el tamaño de la imagen.
Bitmap _bitmapPreScale = BitmapFactory.decodeFile(pathImage);

// no se almacena si es m´sa grande de 800 así que hacemos un resize
int i=2;
int width = _bitmapPreScale.getWidth()/i;
int height = _bitmapPreScale.getHeight()/i;
while(width > 500) {
        i++;
        width = _bitmapPreScale.getWidth()/i;
        height = _bitmapPreScale.getHeight()/i;
}

//Realizamos la transformación.

    Bitmap imagen = BitmapFactory.decodeFile(pathImage);
        Bitmap imagenRetocada = Bitmap.createScaledBitmap(imagen, width, height , false);
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        imagenRetocada.compress(Bitmap.CompressFormat.JPEG, 80, stream);
        byte[] byteArray = stream.toByteArray();

        persona.setImagenB(byteArray);

  //Realizamos la transformación para el thumbail
    ImageView ib = (ImageView) findViewById(R.id.imageViewPersonalData);

  //Realizamos la transformación para el thumbail. Se hace con nuevas variables por el tema de direcciones
//de memoria que si se utiliza la misma la primera pasa a apuntar a la segunda.
    Bitmap imagenRetocadaTh = Bitmap.createScaledBitmap(imagen,ib.getWidth(),ib.getHeight(), false);
        ByteArrayOutputStream streamTh = new ByteArrayOutputStream();
        imagenRetocadaTh.compress(Bitmap.CompressFormat.JPEG, 80, streamTh);
        byte[] byteArrayTh = streamTh.toByteArray();

    persona.setImagenBTh(byteArrayTh);
}
```

Se realiza un resize de la imagen ya que si es muy grande el envío puede fallar recibiendo un fichero tipo Bitmap.

Después se hace una compresión a fichero jpg con una calidad al 80 por ciento de la imagen. Con lo devuelto es lo que ponemos en el campo setImagenB que es lo que recibirá el servidor y guardará en sus ficheros.

Dificultades encontradas

Durante el desarrollo de la aplicación me he encontrado con errores y problemas.

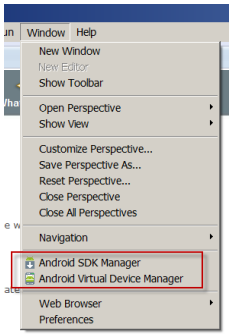
Todos los problemas que se comentan han sido finalmente solucionados o se ha encontrado una forma paralela de desarrollar esa necesidad que por diversos motivos no se podía llevar a cabo. Pero todo ello ha tenido un coste dentro del desarrollo, sobretodo en esfuerzo y tiempo.

1 Instalación Eclipse

Al intentar realizar la instalación del entorno me encontré que había muchas formas diferentes de poder realizarlo, pero escogí la que requería de descargarse el eclipse y descargarse los SDK de forma manual para así poder tener más control sobre todo lo que se ejecuta, instala y debuga.

2 Virtual Device Manager

El segundo problema que me encontré, es que después de haber realizado la instalación y haber programado un par de líneas quería realizar una prueba pero estuve hasta 3 horas investigando que podía pasar para que no apareciese el AVD Manager.



Pude solucionarlo con haciendo que apareciese

El link que daba la solución es:

http://stackoverflow.com/questions/12028811/cant-find-avd-or-sdk-manager-in-eclipse

Para solucionarlo debía:

1. Ir a Window > Customize Perspective (donde el apartado de AVD Manager esta deshabilitado)

2. Se debe habilitar clicando sobre el check.

3 Recoger valores de R desde código

Después de más de tres horas de intentar recoger los valores de R he visto que únicamente con el import basta.

http://stackoverflow.com/questions/2483732/r-id-cannot-be-resolved

Por ejemplo me faltaba poner:

import com.cookbook.simple_activity.R;

Para que pueda tener la aplicación sin errores a la hora de recoger las etiquetas del String:

setContentView(R.layout.game);

También me he encontrado con problemas para aclararme a la hora de recoger los diferentes valores. Una tabla que plasma a la perfección como recoger los datos es :

Table 2.2 How Different Resources Are Referenced from Within Java and XML Files

Resource	Reference in Java	Reference in XML
res/layout/main.xml	R.layout.main	@layout/main
res/drawable-hdpi/icon.png	R.drawable.icon	@drawable/icon
@+id/home_button	R.id.home_button	@id/home_button
<string name="hello">	R.string.hello	@string/hello

4 Apk to code

Fuente de http://stackoverflow.com/questions/3593420/android-getting-source-code-from-an-apk-file

Gracias a estos pasos he podido decompilar el ejemplo del datetime picker para poder introducir el día y la hora en un mismo dialog.

1

Make a new folder and put .apk file in it (which you want to decode). Now rename the extension of this .apk file to .zip (eg.: rename from filename.apk to filename.zip) and save it. Now you get classes.dex files, etc. At this stage you are able to see drawable but not xml and java files, so continue.

2

Now extract this zip apk file in the same folder (or NEW FOLDER). Now download dex2jar from this linkhttp://code.google.com/p/dex2jar/ and extract it to the same folder (or NEW FOLDER). Now open command prompt and change directory to that folder (or NEW FOLDER). Then write `dex2jar classes.dex` and press enter. Now you get classes.dex.dex2jar file in the same folder. Then download java decompiler from http://java.decompiler.free.fr/?q=jdgui and now double click on jd-gui and click on open file. Then open classes.dex.dex2jar file from that folder. Now you get class files and save all these class files (click on file then click "save all sources" in jd-gui) by src name. At this stage you get java source but the xml files are still unreadable, so continue.

3

Now open another new folder and put these files

1. put .apk file which you want to decode
2. download apktool v1.x AND apktool install window (both can be downloaded at the same location) and put in the same folder
3. download framework-res.apk file and put in the same folder (Not all apk file need framework-res.apk file)
4. Open a command window
5. Navigate to the root directory of APKtool and type the following command: `apktool if framework-res.apk`
6. `apktool d "fname".apk` ("fname" denotes filename which you want to decode)
7. now you get a file folder in that folder and now you can easily read xml files also.

4

It's not any step just copy contents of both folder(in this case both new folder)to the single one and now enjoy with source code...

5 Guardar Imágenes en Sql Server 2012

No he podido recuperar una imagen de la BBDD de forma correcta.
Se guarda de forma String o de tipo varbniary , pero a la hora de recuperarla no he podido hacerlo de forma correcta.
Por lo que se realiza un workarround para poder mostrarlas.
Se guarda la imagen en local en la parte servidor y en la BBDD se guarda el path.
Para que cuando se haga la petición recuperará el path y el servidor realizará las operaciones pertinentes para recuperar la imagen y devolverla.
Es un operación más lenta. Pero funciona.

6 Llamadas Rest tipo @Post @Get y parámetros

Llamada al servidor

Para empezar me encontré con el impedimento de que la parte servidor funcionaba correctamente pero desde la petición del móvil o emulador no funcionaba.
El problema residía en que la petición se hacia desde localhost, cosa que desde el terminal no era, así que se cambió la petición al servidor correcto que es la ip del servidor, en mi caso 192.168.1.130-133.

@Post @Get

El guardar la apuesta principal con llamada tipo @Post funciona de forma asíncrona y con llamada normal al método.
Para este desarrollo no he podido averiguar por qué funciona para la llamada de guardar la apuesta principal pero para el resto no, así que todas las peticiones se hacen de forma asíncrona excepto el inserte de la apuesta.

Configuración de Red.

Otro expediente X es que las peticiones REST ya sean @GET o @POST no funcionan en ciertas redes.
Por ejemplo el emulador de Eclipse puede hacer las llamadas de forma correcta, en cualquier red, mientras en el móvil no funciona.
El único sitio donde he podido hacer funcionar el móvil, por defecto, ha sido desde la red de casa qué por suerte es la que más utilizo.

Conclusión

Esto es debido Dependiendo del tipo de conexión que se haya configurado al conectarse, Casa, privada o pública aplicará ciertos criterios de restricción. Para que el servidor pueda recibir las peticiones por el puerto 8080 se debe habilitar en el firewall.

7 Enviar parámetros desde el Fragment

Le he dedicado un tiempo para poder pasar parámetros de una pantalla Fragment a una activity que había sido llamada desde esta última y todo lo estaba haciendo bien, excepto por un pequeño detalle en el Entity.

```
intent.setAction(BetNewActivity.CONTRA);
```

Con esta sentencia establecemos que acción es la que llama, así que en la clase receptora basta con poner:
intent.getAction().compareTo(CONTRA)==0
Para poder reconocer de donde se realiza la llamada y así poder recoger los valores de forma correcta.

8 Imágenes

No he conseguido hacer que la imagen que guardo en la BBDD se recupere de forma correcta. Por lo que se guarda en el directorio local del servidor y se recupera de este mismo directorio. Al adjuntar la imagen dentro de la apuesta, en el momento de guardarla, sin saber por qué se inclina la imagen al mostrarla en el imageView.

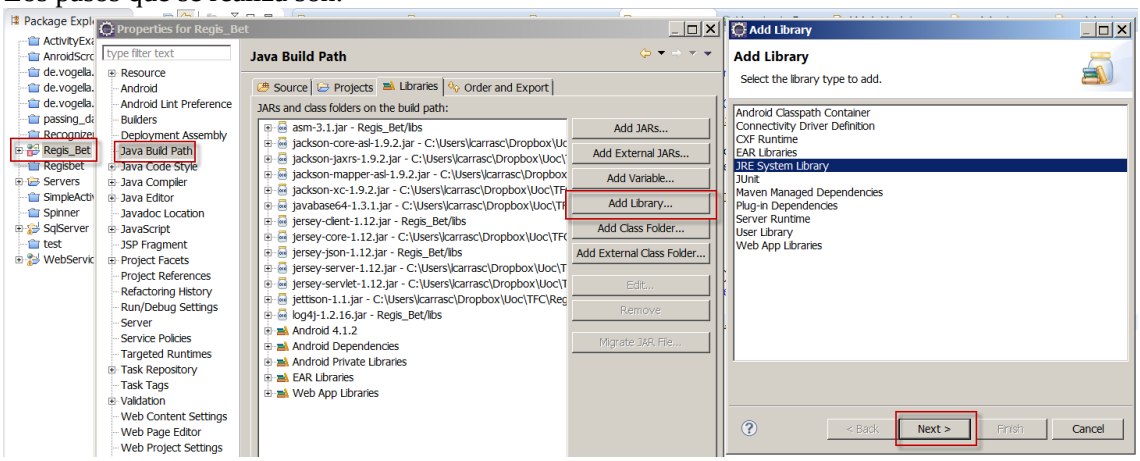
9 Librerías de Java

Al inicial el Eclipse se muestra una serie de errores que en su momento funcionó y sin motivo alguno dejó de hacerlo.
Entre otras clases se muestra el siguiente error:

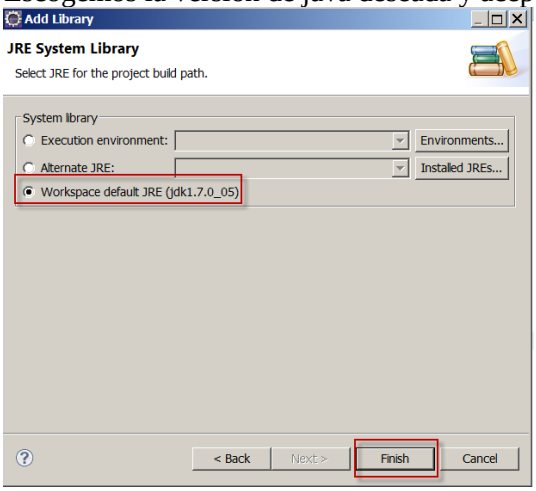
"
Multiple markers at this line
- BASE64Decoder cannot be resolved to a type
- BASE64Decoder cannot be resolved to a type

"

Desde el proyecto de Android no puedo hacer referencias a librerías estándares de java por que les falta añadir la librería. El problema es que cada vez que las pongo cuando reinicio el Eclipse, esta configuración se pierde y siempre tengo que volverla a poner.
Los pasos que se realiza son:



Escogemos la versión de java deseada y aceptamos.



Producto final

Cuando se empezó el proyecto de final de carrera las expectativas de realizar una aplicación con una calidad suficientemente buena como para que saliese al market.

A día de hoy puedo decir tranquilamente que el estado actual de este proyecto no es ni mucho menos el planteado al inicio de la asignatura.

También tengo que decir que el objetivo marcado era muy ambicioso, se planteaba una aplicación con muchas opciones, muy diversas y de una complejidad alta.

A pesar de que hay bastantes carencias con las expectativas iniciales, se ha conseguido realizar una parte del todo operativa y que establece una base para poder seguir desarrollando una aplicación final.

Test Realizados

En la etapa de pruebas se lleva a cabo las pruebas unitarias e integradas de la aplicación.
Se utilizarán dos dispositivos para la realización de las pruebas.

- La máquina virtual de Android que proporciona Eclipse.
- El teléfono Android.

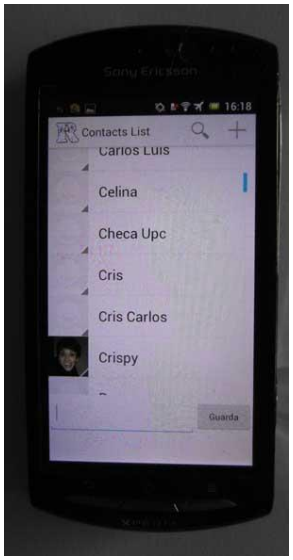
También se entrega el dispositivo móvil Android a una persona que no ha tenido contacto con la aplicación para una prueba completa.

Las pruebas que se han realizado son:
Muestra de apuestas ya existentes.
Navegación por los menús.
Inserción de una nueva apuesta sin foto.
Inserción de una nueva apuesta con foto.
Inserción de una nueva apuesta con las respuestas modificadas.
Inserción de una nueva apuesta con todos los datos insertados.
Inserción de una nueva apuesta con los datos básicos.
Modificar una apuesta existente.
Insertar datos de usuario.
Modificar todos los datos de usuario.

Todas pruebas realizadas se han llevado a cabo con éxito.
En algunas ocasiones no cuando se intentaba escoger una imagen y se volvía atrás la aplicación reportaba un error.

FlashLights

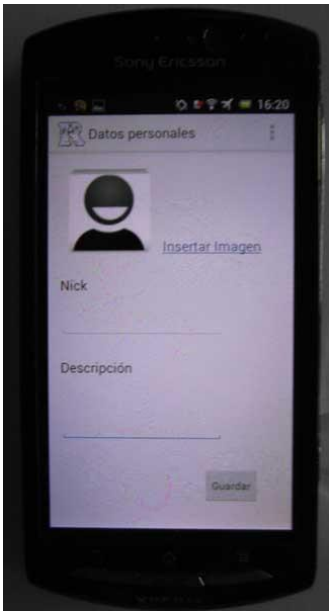
A continuación se muestra algunas capturas de pantalla de la aplicación.

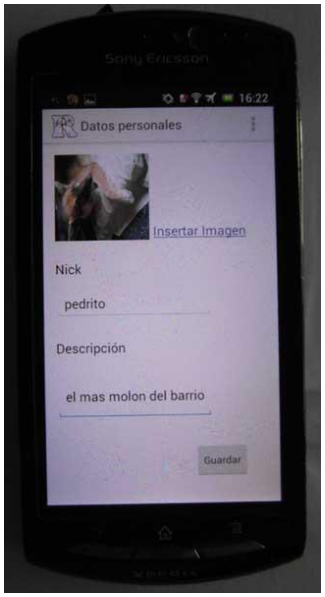



Sony Ericsson
16:19
Nueva Apuesta
Título Apuesta
Completar acción usando
Galería
Gestor de ficheros ASTRO
Xperia™ Galería
Usar modo predet. para esta acción
Editar Respuestas
Guardar
XPERIA


Sony Ericsson
16:19
Nueva Apuesta
Título Apuesta
Set date
Set time
15
16
17
17
18
19
ok
Reset
cancel
Editar Respuestas
Guardar


Sony Ericsson
16:19
Edición de respuestas
Si
No
Guardar


Sony Ericsson
16:20
Datos personales
Insertar Imagen
Nick
Descripción
Guardar


Sony Ericsson
16:22
Datos personales
Insertar Imagen
Nick
pedrito
Descripción
el mas molon del barrio
Guardar

Áreas de mejora

La aplicación puede mejorar de muchas maneras.

Nuevas funcionalidades

Como administración de los roles de usuario en una apuesta y su posterior gestión en la finalización de la apuesta, inserción de vídeo y audio en la apuesta, alarma para cuando la apuesta se acaba, apuestas por PayPal para que el dinero sea real, chat (que es un apartado que me he quedado con muchas ganas de desarrollar).

Funcionalidades existentes

Guardar las imágenes de forma segura dentro de la BBDD, tener cacheadas las imágenes para no consumir recursos innecesariamente, gestión de los adversarios de una forma más clara, el diseño se puede mejorar sobremanera, otorgarle diseño multidispositivo.

Conclusión Final

Desde el inicio del planteamiento de la aplicación se tuvo una visión muy ambiciosa de que es lo que se quería hacer.
Multitud de pantallas, muchas opciones con una gran complejidad en algunos casos.
Partiendo desde la base que no tenía experiencia en el desarrollo de aplicaciones Android, que el tiempo para realizar este desarrollo es limitado y el gran número de pantallas que se debían desarrollar creo acertar al decir que la aplicación cumple una pequeña pero importante parte de la aplicación.
A pesar que se tenía una visión muy optimista, que el planteamiento inicial no era realista en los objetivos del desarrollo se han puesto en práctica los conocimientos adquiridos durante el transcurso de la carrera, se ha asimilado la forma de desarrollar la nueva tecnología y se ha creado una aplicación que cierra un ciclo unitario, que es el de creación de una apuesta, modificación y gestión básica de un usuario.
Como resumen final me atrevo a decir que estoy orgulloso de la aplicación que entrego ya que puedo demostrar como en un corto plazo de tiempo se ha adquirido una buena base para poder continuar desarrollando en esta tecnología.

Bibliografía

BBDD

http://www.sqlite.org/download.html

http://developer.android.com/guide/topics/data/data-storage.html#db

http://developer.android.com/training/basics/data-storage/databases.html

http://www.androidhive.info/2011/11/android-sqlite-database-tutorial/

http://www.androidhive.info/2013/09/android-sqlite-database-with-multiple-tables/

http://www.youtube.com/watch?v=gaOsl2TtMHs

http://www.codeproject.com/Articles/119293/Using-SQLite-Database-with-Android

http://elbauldelprogramador.com/opensource/programacion-android-implementando-un/

BBDD Sql Server

http://theopentutorials.com/examples/java-ee/servlet/how-to-create-a-servlet-with-eclipse-and-tomcat/

https://cwiki.apache.org/confluence/display/STONEHENGE/Configuring+Microsoft+SQL+Server+2008+Express

http://www.aliaspooryorik.com/blog/index.cfm/e/posts.details/post/installing-sql-server-2008-express-162

Configuración BBDD en Eclipse con JDBC

http://msdn.microsoft.com/en-us/library/ms378428.aspx

Creación usuarios en SqlServer 2012

http://www.slideshare.net/narkamo3/habilitar-la-autenticacin-sql-y-crear-un-nuevo-usuario-sql

DataStorage de Archivos

http://developer.android.com/training/basics/data-storage/files.html

Camera interact

http://www.androidhive.info/2013/09/android-working-with-camera-api/

Ejecución en Background

http://dhimitraq.wordpress.com/2012/11/27/using-intentservice-with-alarmmanager-to-schedule-alarms/ -> complicado pero completo.

http://binarybuffer.com/2012/07/executing-scheduled-periodic-tasks-in-android -> Sencillo pero útil, es el escogido.

http://stephendnicholas.com/archives/42 -> complementario al utilizado.

http://danielnadeau.blogspot.com.es/2013/10/android-services-tutorial-run-tasks-in.html#!/

-> Gran tutorial de como hacer background.

Componentes de Presentación Varios

Date Picker

https://code.google.com/p/datetimepicker/

http://developer.android.com/guide/topics/ui/controls/pickers.html

Adapters

http://elbauldelprogramador.com/opensource/programacion-android-interfaz-grafica-2/

Estilos y Temas

http://developer.android.com/guide/topics/ui/themes.html

Common Layouts

http://developer.android.com/guide/topics/ui/declaring-layout.html#CommonLayouts

Súper curso básico de Android (bueno buenísimo para empezar)

http://elbauldelprogramador.com/opensource/programacion-android-recursos-arrays-de/

ListView

http://www.naroh.es/blog/2013/android/listviews-personalizadas-en-android/

http://jarroba.com/listview-o-listado-en-android/

http://www.startcapps.com/blog/tutorial-como-optimizar-una-listview-con-imagenes-de-internet/

Multiselección de ListView

http://www.michenux.net/android-listview-highlight-selected-item-387.html

http://stackoverflow.com/questions/2939767/when-marking-an-item-changing-background-color-in-listview-its-repeating-for

http://stackoverflow.com/questions/6939838/change-the-background-image-of-listview-on-focus

http://www.androidpeople.com/android-custom-listview-tutorial-part-1

Sentencias jdbc

http://www.tutorialspoint.com/jdbc/jdbc-insert-records.htm

Manual Básico

http://www.ntu.edu.sg/home/ehchua/programming/java/JDBC_Basic.html

Tratamiento Imágenes

Guardar imágenes y recuperarlas en BBDD

http://www.java-tips.org/other-api-tips/jdbc/how-to-store-retrieve-image-to-from-sqlserver-2.html -> No funciona correctamente

https://www.inetsoftware.de/products/jdbc-driver/ms-sql/documentation/manual

http://www.cubrid.org/blog/dev-platform/understanding-jdbc-internals-and-timeout-configuration/

http://stackoverflow.com/questions/18599985/send-file-inside-jsonobject-to-rest-webservice

http://www.coderanch.com/t/482256/java/java/Converting-Base-encoded-String-Image

http://www.softwarepassion.com/android-series-get-post-and-multipart-post-requests/

http://www.solutionoferror.com/java/how-to-get-multipartentity-from-httpservletrequest-167635.asp

http://tacticalnuclearstrike.com/2010/01/using-multipartentity-in-android-applications/

Guardar información clave valor con shared preferences

http://developer.android.com/training/basics/data-storage/shared-preferences.html

Principios de desarrollo en Android de entorno móvil

http://developer.android.com/design/videos/index.html

Instalar entorno

http://www.youtube.com/watch?v=aRcb-f3jv4o

Todo lo que me gustaría conocer cuando empecé con Android

http://www.youtube.com/watch?v=h3gPo7qFOFw

Como debugar en eclipse

http://www.youtube.com/watch?v=Zt3RrmP2Xl0

Significados tags de Manifest

http://tools.android.com/tech-docs/tools-attributes

Ejemplos varios de desarrollo

http://examples.javacodegeeks.com/android/core/view/menu/android-actionbar-example/

http://www.javacodegeeks.com/2010/10/android-full-application-tutorial.html

http://www.javacodegeeks.com/tutorials/android-tutorials/android-core-tutorials/

Chat

http://quickblox.com/developers/Android_XMPP_Chat_Sample

http://stackoverflow.com/questions/4543943/how-to-make-a-chat-application-in-android

http://stackoverflow.com/questions/16954712/android-whatsapp-chat-examples good

REST (JAX-RS)

- Part 1 - The Database
- Part 2 - Mapping the Database to JPA Entities
- Part 3 - Mapping JPA entities to XML (using JAXB)
- Part 4 - The RESTful Service
- Part 5 - The Client

http://theopentutorials.com/examples/java-ee/jax-rs/create-a-simple-restful-web-service-using-jersey-jax-rs/