UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO FIN DE CARRERA

Ingeniería de Telecomunicación

Aplicación para movilidad articular y espalda sobre teléfono inteligente iPhone

Óscar Javier Soria Andreu

Octubre de 2015

Aplicación para movilidad articular y espalda sobre teléfono inteligente iPhone

AUTOR: Óscar Javier Soria Andreu

TUTOR: Federico García Salzmann

PONENTE: Eduardo Boemo Scalvinoni

Digital System Lab

Departamento de Tecnología Electrónica y de Comunicaciones

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Octubre de 2015

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Agradecimientos

Quiero agradecer a mi tutor, Federico García Salzmann, y a mi ponente, Eduardo Boemo Scalvinoni, por su ayuda, ya que sin ellos no hubiese sido posible la realización de este proyecto. También a Henar García, la desarrolladora de los textos explicativos y las rutinas de los ejercicios que contienen la aplicación.

A nivel personal, tengo que dar las gracias a mis padres, Luis y Concepción, por su gran apoyo sin el cual no hubiese sido posible la realización de esta carrera ni, por supuesto, de dicho proyecto. No se me puede olvidar dar las gracias a la correctora y traductora Marta Torrelo Altolaguirre, graduada en la Universidad Autónoma de Madrid, por sus consejos a la hora de redactar y revisar esta memoria.

Asimismo, agradezco a todos mis amigos, tanto compañeros de la carrera como a amigos cercanos, por su apoyo a lo largo de todos estos años de esfuerzo y sacrificio. Tengo que nombrar especialmente a Jordi Falco Hernández, ingeniero informático, el cual siempre me ha ayudado en mi formación, me ha animado a seguir adelante y todavía hoy continúa orientándome en mi futuro profesional.

Por último, me gustaría agradecer a la administración de la Escuela Politécnica Superior por su ayuda a la hora de resolver mis dudas en cuanto a temas administrativos, así como a la Università degli Studi di Trento, que durante mi beca Erasmus me ayudaron y apoyaron en todo, contribuyendo a hacer de esta experiencia en Italia un año memorable y enriquecedor a nivel tanto académico como personal.

1. Introducción

La movilidad articular se define como el conjunto de límites y grados de libertad de una articulación en cada plano de movimiento. Esta depende de cuatro elementos: estructura ósea, tendones, ligamentos y músculos. El término de movilidad articular está asociado al de flexibilidad, facilidad para mover partes del cuerpo hasta alcanzar los límites mecánicos naturales de este.

Ambas dependen de parámetros como la actividad física, la edad, el género, o lesiones anteriores.

El aumento moderado de la flexibilidad, aporta claramente grandes beneficios:

- Facilidad para realizar tareas.
- Menor gasto de energía.
- Reducción del riesgo de lesiones musculares (rotura fibrilar y distensiones).
- Mejor coordinación.
- Menos dolores.

Uno de los aspectos positivos de la realización de ejercicios para mejorar la flexibilidad, es que la realización de dichos estiramientos y ejercicios de movilidad no requiere material especial, ni espacio, ni tampoco es violenta o peligrosa. Es adecuada para cualquier edad y sus resultados mejoran rápidamente la calidad de vida. Es decir, es muy accesible, puesto que su práctica no requiere un elevado coste y lo pueden realizar personas de cualquier edad, apreciando notablemente los resultados día a día.

Las pautas generales de este tipo de gimnasia son muy simples:

- Calentamiento.
- Ejercicios con poco peso.
- Movimientos suaves.
- Tratar de maximizar la amplitud de cada movimiento.
- Utilización de una colchoneta en suelo.

En conclusión, una mejora de nuestra movilidad contribuye a mejorar nuestro nivel de vida y se puede realizar mediante ejercicios muy fáciles y sencillos, por lo que

mediante pequeñas guías de texto o de voz se pueden seguir las pautas de forma muy sencilla, lo que resulta ideal tanto para personas con algún tipo de discapacidad como para aquellas que sufren pequeñas molestias y desean mejorar su movilidad.

1.1 Motivación

Uno de los aspectos principales que me han llevado a la realización de dicho proyecto es mi pasión por los deportes. Como amante del fútbol o del gimnasio, el hecho de realizar una aplicación que ayuda a mejorar en este caso la movilidad articular y flexibilidad de una persona, fue una de mis grandes motivaciones para decantarme por este proyecto.

Anteriormente si se necesitaba consultar el correo, leer un documento, leer un libro, acceder a cualquier tipo de información o llamar por teléfono se necesitaba un dispositivo para cada función pero con el uso de las aplicaciones hace que todo esto sea posible en un simple dispositivo. Cada aplicación permite hacer cosas muy concretas, por lo que hoy en día todos los usuarios que utilizan móviles descargan y usan aplicaciones para hacer su vida más fácil en consecuencia estas aplicaciones son de vital importancia y utilizadas por una cantidad muy grande de usuarios.

Otro punto que es preciso tener en cuenta es que la aplicación realizada contiene una versión con ayudas de voz. Esto supone una gran ayuda para los invidentes, ya que presenta un menú totalmente rediseñado y muy fácil de manejar. Además, todo este guiado sigue la técnica de lectura fácil con instrucciones muy sencillas y bien dirigidas.

1.2 Objetivos

El DSLab de la Universidad Autónoma de Madrid desarrolló una aplicación para Android de estiramiento muscular destinada a personas ciegas. En cambio, el objetivo de este proyecto es el desarrollo de una aplicación en iOS que reproduzca la información de dicha aplicación. Se ha llevado a cabo la adaptación de la aplicación, así como la reprogramación, utilizando las distintas herramientas que ofrece Apple y modificando su funcionamiento en la versión con ayudas de voz.

Puede descargarse la versión de Android a través del siguiente enlace: <https://play.google.com/store/apps/details?id=com.movilidadarticularyespalda&hl=es>

Los objetivos técnicos pueden resumirse en los siguientes puntos:

- Desarrollar y poner a punto una versión principal con ayudas por voz. Debe estar pensada para aquellas personas que tienen problemas de visión. Esta debe disponer de una interfaz de usuario basada en gestos, utilizando además conceptos de lectura fácil. Las instrucciones de los ejercicios deben ser pregrabadas o leídas por el sintetizador de voz del teléfono.
- Desarrollar y poner a punto de la versión llamada clásica. Este deberá ser más visual y con mayores opciones, pensada para aquellas personas que no presentan ninguna discapacidad visual.
- Publicación en el Apple Store.
- Ejercicios con una fácil implementación.

Los objetivos de este proyecto a nivel educativo son:

- Manejo de las herramientas Apple
- Aprendizaje de Objective-C
- Aprendizaje del proceso de certificación y publicación de una app en Apple.
- Verificación exhaustiva de una aplicación y creación de un producto acabado; no una versión prototipo.
- Desarrollo de la memoria final del proyecto

Como se ha comentado anteriormente, el objetivo principal de este tipo del trabajo es mejorar la vida y el bienestar de las personas con discapacidad visual y que la información sea accesible para todos.

Por último, es preciso comprobar y testear el correcto funcionamiento de la aplicación mediante pruebas continuas sobre diferentes dispositivos, aunque siempre basándose en su aplicación homóloga en Android.

1.3 Organización del trabajo

La organización de este trabajo se ha llevado a cabo a través de la división de la memoria en capítulos, que corresponden con cada una de las fases generales de desarrollo del proyecto:

1) **Capítulo 1.- Introducción:** se explica brevemente la razón de ser del presente proyecto, los pasos a dar para la ejecución del mismo objetivo y la organización de este informe.

2) **Capítulo 2.- Estado del arte:** se hace una revisión sobre todos aquellos recursos que han sido implementados antes de la realización de este proyecto, contribuyendo así a mejorar el día a día de las personas. También se explicará qué trabajo se ha realizado previamente con respecto a los temas que se tratan, es decir, aplicaciones móviles, sistemas de guiado de voz y lectores de pantalla dirigidos a personas con discapacidad visual.

3) **Capítulo 3.- Diseño:** plantea a fondo el problema al que se hace frente en esta memoria, a través de la cual se tratará de obtener soluciones para realizar el prototipo, facilitando al final una idea general esquemática de la aplicación y sus apartados. Se compara su diseño al de la aplicación homóloga en Android.

4) **Capítulo 4.- Desarrollo**: se explica con detalle los pasos que se han seguido a la hora de desarrollar el código de la aplicación.

5) **Capítulo 5.-Pruebas y resultados:** se analizan las pruebas realizadas. Se tratará de comprobar el manejo de la aplicación con una persona discapacitada visualmente.

6) **Capítulo 6.- Conclusiones, aprendizaje y trabajos futuros:** se abordan las conclusiones finales del diseño. Además, se explican los conocimientos adquiridos en el tiempo de realización del proyecto y, por último, los trabajos de mejora de este.

2. Estado del arte

2.1 Teléfono inteligente

Se trata de un teléfono móvil que contiene una plataforma informática con capacidad de almacenamiento de datos, realización de múltiples actividades, semejante a un miniordenador y con múltiples opciones de conectividad, por lo que supone un gran avance respecto a un teléfono convencional. Con un teléfono inteligente se pueden hacer múltiples tareas al mismo tiempo, como consultar el correo electrónico, revisar la agenda, hacer llamadas, ver vídeos e incluso sincronizar un dispositivo con otros o enviar datos, todo ello sin necesidad de interrumpir ninguna de las tareas. Es prácticamente como tener un ordenador de bolsillo y puede incluso llegar a sustituir a un ordenador personal.

Por todo ello, se puede apreciar que un *smartphone* es un teléfono móvil pero mucho más potente que los primeros que se comercializaron y, evidentemente, está destinado a ser utilizado por aquellas personas que necesitan tener acceso permanente a su correo electrónico o a Internet, como es el caso de los empresarios, e incluso aquellas cuyo uso desmesurado ha provocado una gran adicción.

Sin embargo, estos dispositivos pueden ser vulnerables a virus y ataques al SO, tal y como sucede en la actualidad con los equipos portátiles o de sobremesa. Existen diversos diseños y cada uno está adaptado a un tipo de público. Se tienen en cuenta factores como el diseño de la tapa, el diseño del teclado, su estética y un sinfín detalles con tal de agradar al usuario.

A finales de 2014, los teléfonos inteligentes en España fueron utilizados por el 53.7% de la población española a partir de 15 años y el número de usuarios sigue aumentando cada año.

En definitiva, se llega a la conclusión de que el uso de estos teléfonos mejora nuestra calidad de vida y se hacen indispensables en el día a día, puesto que tienen un sinfín de funciones y ventajas con respecto a sus predecesores.

2.2 Sistemas Operativos en teléfonos inteligentes:

Los sistemas operativos que existen actualmente en el mercado son los siguientes:

1) Android

Sistema operativo basado en Linux, diseñado para pantallas táctiles de tabletas o móviles. Fue desarrollado por Android contando con el apoyo de Google. El primer móvil con sistema Android fue desarrollado en 2008 y presenta una cuota de mercado de más del 80%. Sus aplicaciones son desarrolladas en Java y se necesitan plataformas como Eclipse para desarrollar aplicaciones en este sistema. En Google Play se dispone del catálogo de las aplicaciones que se pueden descargar. El código de Android es abierto. Google liberó Android bajo licencia Apache, con lo cual cualquier persona puede realizar una aplicación para Android. El sistema Android puede hacer funcionar a la vez varias aplicaciones y gestionarlas, dejarlas en modo suspensión si no se utilizan e incluso cerrarlas si llevan un periodo determinado de inactividad. De esta manera se evita un consumo.

2) iOS

Sistema operativo de la empresa Apple Inc. desarrollado para iPhone. También se utiliza en iPod, iPad y Apple TV. Destaca por su interfaz gráfica y su interactividad mediante gestos. Se trata de un sistema operativo derivado del propio OS que se utiliza en los ordenadores Mac, cuyos dispositivos son de gran calidad. Los elementos de control se componen de deslizadores, interruptores y botones. La respuesta a las órdenes del usuario es inmediata y provee una interfaz fluida y vistosa con una gran calidad de imagen. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen funciones diferentes dependiendo del contexto de la interfaz y de la herramienta que se esté utilizando. Sus dispositivos son los segundos más utilizados en el mundo y tecnológicamente son bastante avanzados.

Para poder desarrollar una aplicación es necesaria una licencia de desarrollador y se puede programar en Objetive-C utilizando Xcode.

3) BlackBerry OS

Sistema operativo desarrollado por Reserach in Montion para dispositivos BlackBerry. Se trata de un sistema operativo con características actualizadas a la última tecnología. Sistema que permite multitarea y soporta diferentes métodos de entrada adoptados por RIM para su uso en computadoras de mano. Permite sincronización con programas como Microsoft Exchange y acceso al correo electrónico y permite hacer las funciones usuales de un teléfono móvil.

Tiene tres versiones con QNX: V6, V7 y V10. En las versiones seis y siete se puede programar en Java y en la V10 ya se puede programaren C, C++, Adobe AIR, Java e incluso HTML5+JavaScript+CSS.

4) Windows Phone (Microsoft)

Es un sistema operativo desarrollado por la compañía Microsoft, como sucesor de la plataforma Windows Mobile. Está enfocado a un público general no como su predecesor, que estaba enfocado a un público empresarial. Busca competir con sus rivales Android y iOS e integra varios de sus servicios propios como Onedrive, Skype y Xbox Live en el sistema. Windows Phone y Windows Mobile no son compatibles, con lo que todas las aplicaciones que existían de Windows Mobile se quedaran obsoletas con el paso del tiempo.

En noviembre se producirá el lanzamiento de un nuevo sistema operativo que se ha desarrollado para móviles Windows 10 Mobile, con lo que pretenden ser una alternativa viable que genere una mayor cuota de mercado.

5) Symbian

Sistema operativo desarrollado por una alianza de empresas de telefonía móvil Nokia, Sony Psion, Samsung, Arima, Benq, Fujitsu, LG, Motorola, Mitsubishi, Panasonic, Sharp, etc. cuyo objetivo era el de crear un sistema móvil que pudiera competir con Windows Mobile, Android de Google y iOS de Apple.

En enero de 2013, Nokia anunció que el Nokia 808 era el último modelo de la compañía con este sistema operativo, y por tanto, el último teléfono con Symbian.

6) BADA (Samsung)

Sistema operativo para móvil desarrollado por Samsung y actualmente reemplazado por Tizen. Está diseñado para móviles de gama alta y de gama baja. En noviembre de 2010 se anunció la plataforma BADA y más tarde el WAVE S8500 fue el primer móvil con BADA. Las aplicaciones son desarrolladas en C++. Este sistema operativo no tuvo mucho éxito. No se pueden instalar aplicaciones fuera de la tienda y tampoco permite el uso de ningún tipo de programa de VoIP/SIP ni el uso de Whatsapp lo que hace inviable este SO.

Cuota de mercado en España

En España los dispositivos móviles están claramente dominados por Android que ocupa un 87.6% de cuota de mercado, mientras que iOS tiene un 8.7%. Aunque se está produciendo un aumento de usuarios de iOS año tras año mientras que los usuarios de Android bajan pero es un cambio muy lento. España es de los pocos países europeos que tiene una cuota de mercado para iOS menor del 10%.

La gran diferencia proviene del hecho que iOS no posee dispositivos de gama baja o media, lo que hace que sea una marca más exclusiva que el resto y no sea accesible para todos.

2.3 Problemática del mundo de la discapacidad visual:

Actualmente, la informática se ha convertido en una vía de acceso de vital importancia en la sociedad de la Información. Hasta hace poco, una persona con ceguera no podía acceder a cualquier tipo de información, pero afortunadamente esto ha sido resuelto gracias a los avances tecnológicos. De este modo, hoy en día una persona con discapacidad visual puede leer un periódico por internet, leer un libro ayudado por un sistema de guiado de voz, trabajar a distancia, etc. lo que ha supuesto un sinfín de mejoras en su calidad de vida.

Ampliadores de Pantalla:

Las personas con baja visibilidad poseen un remanente visual que les permite ver algunos elementos en el monitor del ordenador, pero no pueden diferenciar los detalles más pequeños. Estos programas aumentan el tamaño de las imágenes o caracteres de dos a cincuenta veces para que puedan diferenciarlos mediante el uso del teclado o el ratón.

Lectores de pantalla:

Se trata de programas que se comunican con el usuario del dispositivo u ordenador, quien recibe una descripción en voz sintética por los altavoces o parlantes del dispositivo u ordenador. No se necesitan programas diferentes, sino que mediante este software de lector de pantalla se pueden utilizar los mismos programas que utilizaría cualquier usuario. El uso de estos programas se realiza mediante comandos de teclado.

Terminales con lectura Braille:

Envían la información contenida en la pantalla hasta el usuario utilizando caracteres braille dispuestos en una línea de 20 a 80 ocurrencias, líneas que el usuario lee al tacto. También cuentan con un teclado propio para realizar todas las funciones de lectura modificación o escritura de los contenidos. Estos equipos suelen tener un elevado coste, aunque son una gran solución para gente que tenga visibilidad cero.

Impresoras Braille:

Impresoras que permiten imprimir por una cara textos en braille. Estas suelen ser de velocidad lenta pero son necesarias, puesto que hasta hace poco la única manera de escribir en braille era a través de una regleta o de una máquina de escribir braille.

Dificultades de accesibilidad:

Estos problemas se reducen en dificultades con el software, dificultades con el hardware y dificultades en la navegación por Internet.

Aunque haya programas de lectura, existen muchos problemas de accesibilidad debido a la interfaz gráfica compleja de estos programas. Otro de los grandes problemas es el manejo con los diferentes botones, interruptores, ratones, teclado, etc., que permiten el manejo del ordenador. Algunos de estos problemas se pueden resolver utilizando programas que controlen los dispositivos pudiéndose apagar, encender y regular todos los componentes físicos del ordenador desde el programa.

Algunos diseños de las páginas web dificultan la interpretación de los programas lectores de pantalla puesto que no pueden describir bien las imágenes ni interpretar diseños complejos. Por ejemplo, si una página nos envía a una foto solo leerá el nombre de la foto y no dará más información.

Otro problema es la difícil interpretación de hipervínculos o tablas dentro de la navegación. Por último, hay webs que utilizan elementos antirobots y nos dan una clave en forma de imagen. Es muy difícil de interpretar, aunque algunas incorporan una clave por voz para evitar este tipo de problemas.

2.4 Avances médicos-tecnológicos

Existen dos grandes posibles curas: un chip implantado en la retina y la regeneración de las córneas dañadas mediante el uso de células madre de adultos.

A continuación, se explicarán brevemente amabas técnicas.

2.4.1 Visión biónica

Esta tecnología denominada por sus creadores como "un avance sin precedentes en cuanto a prótesis electrónicas para la visión", fue implantada por los científicos alemanes del instituto Retinal Implant AG y el llamado instituto para investigación oftálmica, the Institute for Ophthalmic Research, de la universidad de Tuebingen.

El proceso consiste en la implantación de un chip subretinal que devuelve al paciente la visión tanto de formas como de objetos.

Los resultados de este avance mostraron que un 27% de los pacientes obtuvieron resultados óptimos, ya que eran capaces de identificar e ubicar objetos en una mesa, caminar, acercarse a una persona en concreto, consultar la hora o incluso distinguir entre varios tonos de grises.

No obstante, también es cierto que aquellos sujetos con una discapacidad visual avanzada obtuvieron resultados escasos con esta técnica.

Esta técnica ha contribuido, sin duda, a ayudar a un gran número de personas que sufren la llamada retinitis pigmentosa, enfermedad degenerativa que provoca pérdida de visión nocturna, visión de túnel y finalmente la ceguera total.

En definitiva, esta técnica ha ayudado en gran medida a avanzar en la medicina biónica, aunque también es cierto que todavía queda mucho por recorrer para conseguir perfeccionar muchas de estas técnicas. Es en este punto donde se plantea la siguiente cuestión: Si se continúa avanzando en las técnicas más avanzadas, ¿es posible que en unos años la ceguera pueda curarse sin problema alguno?

2.4.2 Regeneración de la córnea

Los investigadores del Eye and Ear/Schepens Eye Research Institute, delBoston Children's Hospital, del Brigham and Women's Hospital y del the VA Boston Healthcare System consiguieron una vía para mejorar el crecimiento de tejido de la córnea humana. De este modo, consiguieron regenerar un tejido humano a partir de células madre humanas adultas"

Como bien explica Frank, uno de los desarrolladores de este descubrimiento, Frank, afirma que "la clave está en el uso de una molécula conocida como ABCB5, que actúa como un marcador para las ‘escurridizas’ células madre del limbo".

Los primeros ensayos se realizaron en ratones, pero pronto el equipo se dio cuenta de que este método podría aplicarse a humanos, lo que les permitiría empezar las primeras pruebas en estos.

Este hallazgo permitirá restaurar la córnea para, de este modo, restaurar la visión a las personas afectadas por todo tipo de problemas relacionados con la córnea.

2.5 Aplicaciones móviles para personas con discapacidad visual

El avance de la tecnología ha hecho posible la creación de herramientas como el reconocimiento automático del habla o sintetizadores de voz a texto lo que facilita el uso de dispositivos móviles a personas invidentes. Estas tecnologías se han ido adaptando y actualizando para conseguir un resultado óptimo que ha permitido un aumento en la calidad de vida del usuario.

Hace poco se presentó un móvil, el modelo B-Touch de Zhenwei You, que incluye pantalla táctil-Braille, sistema de reconocimiento de voz, escáner de objetos y lector de textos.

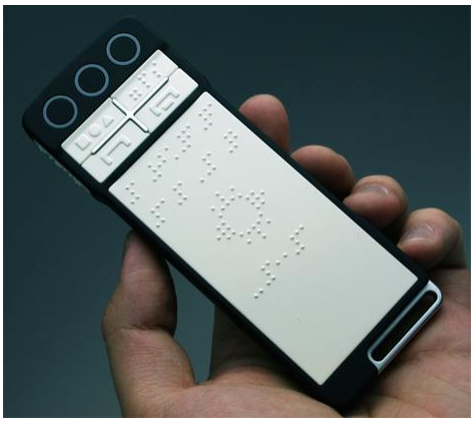


Figura 1 - B-Touch de Zhenwei You

Otro ejemplo de teléfono móvil especial para discapacitados es el Owasys 22C que es el primer teléfono lanzado en España y Europa con funciones especiales para discapacitados visuales . Este teléfono comunica al usuario a través de mensajes de voz las llamadas perdidas, los mensajes recibidos, el estado del teléfono (batería, hora, fecha, cobertura, etc.) o el contenido de los mensajes antes de enviarlos.

Figura 2 - Owasys 22C

Todos los SO están intentado incorporar herramientas para su accesibilidad como Apple, Android o Symbian que cada vez desarrollan más y mejores herramientas que sirven de gran ayuda a los discapacitados visuales.

Aplicaciones:

Google, uno de los pioneros y grandes investigadores en este campo ha creado una aplicación que permite manejar el móvil sin necesidad de mirar la pantalla, lo que supone una gran ayuda para discapacitados y personas al volante. Google actualmente trabaja en "screen reader" software libre para Android y en nuevos desarrollos para pantallas táctiles.

Por otro lado Vodafone cuenta con Easy Walk y Vodafone Speak. Easy Walk es un servicio de localización GPS que transforma los mensajes de texto en voz y con lo cual permite a la persona con discapacidad saber su localización y Vodafone Speak es una aplicación basada en Symbian que permite acceder a toda la información del teléfono mediante un lector de pantalla que transforma la información en mensajes de voz.

EL CIDAT Centro de Investigación, Desarrollo y Aplicación Tiflotécnica ha creado nuevos revisores de pantallas "Mobile Speak Pocket" para PDAs y el "Mobile Speak Smartphones" para teléfonos inteligentes.

La compañía iVisit está trabajando en una tecnología llamada "SeeScan" que consiste en usar el móvil como un escáner para poder leer códigos de un producto y una voz indica el nombre y características del producto. Esta tecnología estará disponible a finales de año para iPhones y Symbian.

Otro proyecto es el SlimVoice que pretende desarrollar un API que aportará una capa de accesibilidad a los nuevos modelos de teléfonos la cual dará soporte para síntesis de voz, utilización de lectura braille y adaptación de la interfaz visual para personas con discapacidad visual.

2.6 Herramientas de accesibilidad que nos proporciona Apple

En la página web de Apple se explican los recursos que la compañía ofrece para ayudar a las personas con discapacidad visual.

Estas son las diferentes opciones que nos ofrece:

1) VoiceOver

VoiceOver es un lector de pantalla con el cual es posible desplazarse en la pantalla del dispositivo mediante una serie de gestos. Esto hace posible la navegación en los menús existentes tanto para iPad, iPhone y iPod Touch. También se puede establecer su velocidad de habla.

2) Leer Pantalla

Con esta herramienta es posible leer el contenido de una pantalla reproduciendo su contenido de forma sonora, haciendo posible la consulta de emails, páginas webs, libros u otro tipo de elementos que se quieran leer. Hay que activar este lector desde ajustes del dispositivo y desplazar los dedos hacia abajo desde la parte superior de la pantalla para que lea el contenido de esta. También posee la opción de configurar el idioma, la velocidad del habla y la elección de palabras que se quieran resaltar.

3) Siri

Siri es el ayudante inteligente de Apple. Puede realizar llamadas, enviar mensajes, programar reuniones e habilitar y desactivar VoiceOver, Invertir Colores e Acceso Guiado. Siri está integrado con VoiceOver, con lo que se puede realizar preguntas acerca de cualquier cosa, como dónde está la parada de autobús más próxima, y se obtiene la respuesta a dicha pregunta en voz alta.

4) Dictado

En el teclado de los dispositivos se dispone de un botón en forma de micrófono al pulsarlo recoge nuestro mensaje de voz y lo transforma a texto pudiéndose así escribir mensajes, correos, notas o direcciones de páginas web que se deseen.

5) Zoom

Zoom es un ampliador de pantalla que funciona en todos los menús y aplicaciones de iOS, desde Mail y Safari hasta las pantallas de inicio y desbloqueo. Además, es compatible con todas las apps del App Store. Se puede ajustar el zoom entre un 100 y un 1.500%, y acceder a varias opciones de filtro en cualquier modo. Es compatible con todos los gestos ya conocidos para navegar por el dispositivo y también es compatible también con VoiceOver para poder ver y escuchar todo lo que hay en pantalla.

6) AVSpeechSynthesizer

Es el sintetizador de voz texto a discurso que se ha utilizado en este proyecto. Su función es leer las acciones de cada ejercicio. En él se puede ajustar la velocidad de habla, el idioma y la pronunciación.

7) Compatibilidad con dispositivos braille

El iPhone, iPad y iPod Touch son compatibles con más de 40 dispositivos braille con Bluetooth. Por lo que se pueden conectar y seguir utilizando el dispositivo sin tener que instalar ningún software adicional. Además los dispositivos iPhone, iPad y iPod Touch incluyen tablas de braille para más de 25 idiomas.

2.7 Importancia de los estiramientos:

Aunque se ha comentado en la introducción algunos de los beneficios que conllevan los estiramientos, es conveniente desarrollar con mayor profundidad este tema ya que es una parte fundamental del proyecto.

La gente tiende a no realizar estiramientos porque lo consideran una pérdida de tiempo. Sin embargo, estos son necesarios y producen beneficios físicos.

Al realizar estiramientos se consigue evitar lesiones y problemas musculares. La realización continua de estos ejercicios ayuda a hacer actividades con menos esfuerzo y con mayor eficacia. Gracias a todo esto se conseguirá una postura corporal óptima.

Una amplitud de movimiento en las articulaciones proporciona una mejora de movilidad y un riesgo menor de lesiones. Los estiramientos hacen que aumente la circulación y el aporte de nutrientes a través de la sangre y así se reduce el espesor de líquido sinovial, reduciendo la posibilidad de padecer enfermedades degenerativas en las articulaciones.

Otro punto a tener en cuenta es que los estiramientos pueden ayudar a la recuperación de una lesión e incluso aliviar el dolor. Esto se atribuye a la mejora de la temperatura en los músculos, al aumento del aporte sanguíneo y mejora de la circulación que conllevan la realización de los mismos.

Un calentamiento antes de una actividad física intensa es esencial y ayudará a mejorar la realización de esta actividad. Tienen que ser estiramientos sencillos y nunca deben alcanzar un nivel de tensión extrema. No se deben realizar estiramientos si se acaba de sufrir una lesión, fractura o si se tienen dolores punzantes.

Por último, los estiramientos realizados en un buen ambiente contribuyen a una relajación muscular y mental dando una sensación de satisfacción personal para así poder liberarse de la presión del día a día.

2.8 Estudio de público objetivo:

En el apartado en el que se habló de los teléfonos inteligentes se comentó que a finales de 2014 los teléfonos inteligentes en España fueron utilizados por el 53.7% de la población española a partir de 15 años. Por ello, calculando el número de personas mayores a 15 años se obtendrá el número de personas que utilizan teléfonos inteligentes. Según el Instituto Nacional de Estadística el número de personas mayores de 15 años es igual a 39.619.299 personas. Como conclusión, se establece que 21.275.564 personas utilizan teléfonos inteligentes. Se da por hecho que las personas que practican deporte y tienen interés en los estiramientos tienen más de 15 años.

De estas personas, el 8.79% usa iOS por lo que los que utilizan iPhones son 1.870.122.

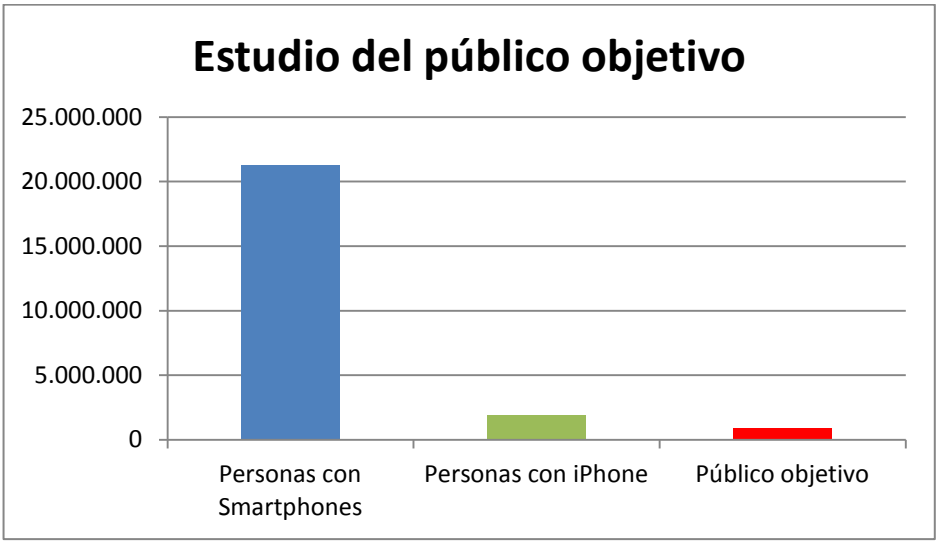


Tabla 1 - Estudio del público objetivo

A continuación se buscará el porcentaje de la población que hace deporte, puesto que todo el mundo que hace deporte necesita realizar estiramientos. El 46% de los españoles hace deporte de forma regular, por lo que los clientes potenciales son 860.256 personas.

Lógicamente, el número de personas que se descargará la aplicación será muy reducido con respecto a este número, ya que aunque se deben hacer estiramientos por las razones comentadas en el apartado anterior, un número muy reducido realiza estos ejercicios.

3. Diseño

3.1 Introducción

La programación de dispositivos Apple no es técnicamente más compleja que la equivalente en Android, pero resulta mucho más difícil encontrar información y códigos de ejemplo ya implementados, lo que dificulta en gran medida el trabajo que se lleva a cabo. Por un lado, hay menos penetración de teléfonos con este SO, puesto que son aparatos caros y su programación requiere un ordenador Apple. También hay que tener una licencia de programador en este caso proporcionado por el DSLab, que tiene estatus de desarrollador y mantiene una licencia activa donde debe enrolarse cada miembro del laboratorio.

El diseño se basa en la aplicación ya realizada para Android, por lo que a nivel visual se parecen pero el código se ha creado de cero. Se han utilizado recursos como un curso de desarrollo de aplicaciones para iOS 8, el foro de desarrolladores de Apple, que facilita mucha información, así como la consulta de varios libros de biblioteca de la escuela.

Como se ha comentado anteriormente, para poder comenzar a programar en Apple se necesita:

- Ordenador Mac donde se desarrollara el código
- Dispositivo de pruebas, en mi caso un iPhone 4s
- Xcode 7.0 que admite la versión actual del iOS (iOS 9)
- Licencia del programa de desarrolladores de Apple

Lo cierto es que son unas exigencias altas, lo que dificulta su desarrollo, ya que el volumen de información en comparación con Android es bastante reducido, por lo que se avanza más lento y resulta mucho más complejo aprender a programar en este entorno.

Para programar se ha utilizado Xcode, herramienta de Apple que a día de hoy es la única viable para la realización de aplicaciones móviles.

Existen dos lenguajes de programación para el desarrollo de aplicaciones: Objective-C y Swift. Se trata de dos lenguajes similares. En este caso, se optó por Objective-C porque es el que aporta más información y mayor cantidad de ejemplos ya realizados. Cuando se comenzó a realizar este proyecto acababa de ser lanzado el lenguaje Swift, con lo que se consideró más fiable utilizar Objective-C.

Las aplicaciones se basan en vistas, que son las pantallas que muestra el dispositivo, por lo que es necesario utilizar UIStoryBoards y las diferentes opciones que nos dan estas vistas. Los menús vienen ya establecidos en la aplicación realizada para Android, así que se han transformado con la interfaz que ofrece Xcode para que tengan un aspecto similar. Las vistas vienen controladas por sus UIViewController y cada una tiene una archivo .m donde se define el funcionamiento y un archivo .h donde se define los componentes de la vista.

En la siguiente figura se observa la estructura del programa Xcode:

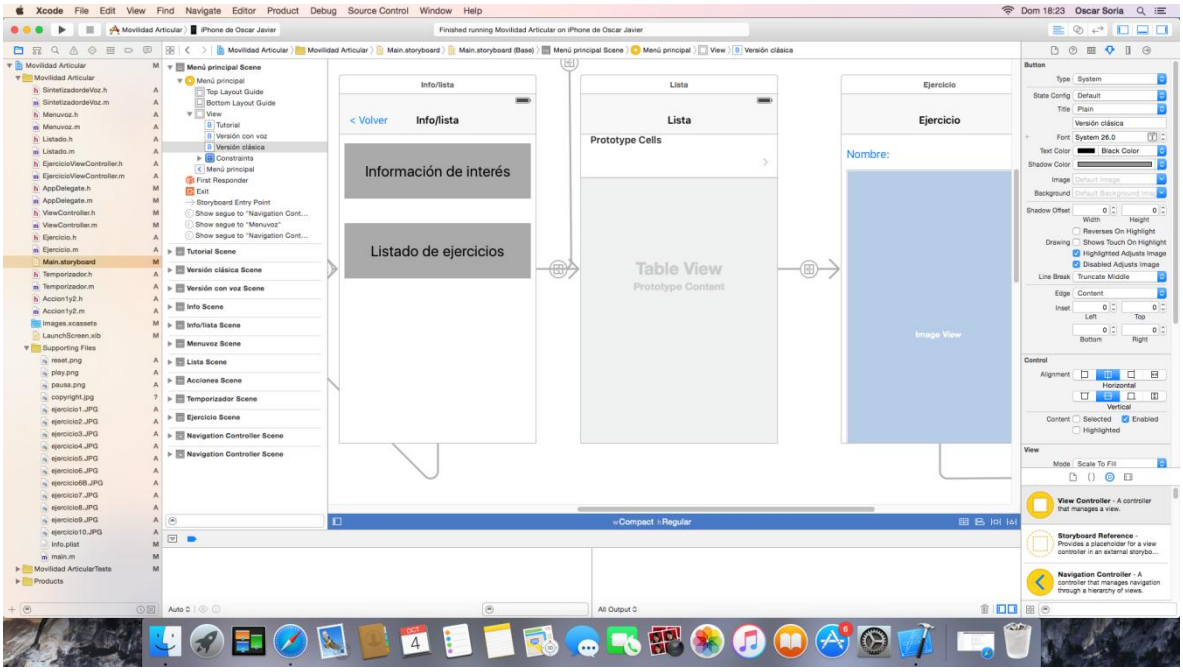


Figura 3 - Interfaz de Xcode

a) La barra superior se compone de:

- Botón para ejecutar el código
- Boton para parar la ejecucion del código
- Seleccionador para indicar donde ejecutar el código

- Cuadro de texto donde se muestra el estado de nuestro código y si se han producido errores o avisos a la hora de elaborarlo.
- Botones en la parte derecha que ocultan o muestran las diferentes zonas del Xcode.

b) En la zona izquierda se muestra los archivos en que se divide el proyecto. En esta zona tambien se puede cambiar las opciones de compilación, la orientacion y todo este tipo de ajustes

c) En la zona central se muestra el contenido de los archivos y es donde se elaborará el código. En el caso de la imagen anterior se muestra el *main stroryboard.*

d) En la zona derecha al seleccionar un elemento como un botón , vista, tabla, *scroll*er, etc., aparecen sus opciones de configuración. Por ejemplo, en un botón se puede modificar el tamaño, letra , color, conexiones, posición, etc.

e) En la zona inferior cuando se compila aparecen mensajes de estado, marcas que se hayan puesto para mostrar un resultado parcial (NSLog) y errores o avisos que se han producido.

f) En la zona inferior derecha se muestra nuestro UIKit que son los elementos que podemos utilizar en la vista. Por ejemplo en la aplicación se utiliza UIButton, UIScrollView, UILabel, UITableViewController, UINavigationController, UIView, UIItemBarButton, etc.

Una vez presentado el entorno de programación, se detallará la estructura del proyecto y se explicará cómo se ha implementado, detallando todo su contenido.

3.2 OS X Yosemite

Para poder utilizar la versión Xcode 7.0 es necesario tener como mínimo la versión 10.10.4 del OS X Yosemite. Es necesario tener instalado Xcode 7.0 puesto que es el que incorpora la compatibilidad con el nuevo y actual iOS 9. La versión actual de este sistema operativo es la 10.10.5.

Figura 4 - OS X Yosemite

Yosemite soporta todos los Mac capaces de ejecutar OS X Mavericks, pues se requieren 2 GB de RAM, 8 GB de espacio disponibles y OS X 10.6.8 o posterior. Sin embargo, se recomienda disponer de Bluetooth 4.0.

Producto	Modelo
iMac	(mediados de 2007 o más reciente)
MacBook	(finales de 2008 de aluminio, o principios de 2009 o más reciente)
MacBook Pro	(mediados / finales de 2007 o más reciente)
MacBook Air	(finales de 2008 o más reciente)
Mac mini	(principios de 2009 o más reciente)
Mac Pro	(principios de 2008 o más reciente)
Xserve	(principios de 2009)

Tabla 2 - Evolución histórica de los ordenadores Mac

Yosemite presenta una gran modificación del interfaz del sistema operativo, el cual fue diseñado inspirándose en la apariencia de iOS 7 pero manteniendo la metáfora de escritorio de OS X. Se han incluido más cambios notables en el diseño como nuevos iconos, combinaciones de colores claros y oscuros, y el remplazo de la fuente predeterminada del sistema de Lucida Grande a Helvetica Neue.

3.3 Estructura del proyecto

Para la realización de este proyecto, se ha utilizado una programación basada en UIStoryboards, con Automatic Reference Counting (ARC) y Autolayout. El ARC hace que el manejo de la memoria lo realice el propio compilador y es obligatorio utilizarlo.

Los Autolayout son una forma de hacer que nuestra vista se vea igual, independientemente del dispositivo que se utilice, mediante el uso de una serie de constantes que se explicarán más adelante. Los elementos que se utilizan son los pertenecientes a la familia UIKit y son los botones, campos de texto, *scrollers*, etiquetas, etc que se utilizan en nuestros *storyboards*.

La aplicación consta de tres apartados: una versión visual con información e imágenes, otra con guiado de voz y un tutorial que explica el funcionamiento de ambas.

Esta es la estructura general del proyecto y porque vista (UIViewController) está controlado cada parte del proyecto. De este modo, la estructura es la siguiente:

- Menú principal controlado por el primer *view controller*
- Tutorial controlado por el primer *view controller*
- Menú secundario con información de interés y lista de ejercicios.
    - Información de interés controlada por el primer *view controller*
    - Lista controlada por el *view controller* llamado listado.
    - Cada ejercicio se descompone en imagen, acciones y temporizador
        - Menú del ejercicio controlado por el *view controller* ejercicio
        - Acciones controladas por el *view controller* acciones
        - Temporizador controlado por el *view controller* temporizador
- Menú de versión con voz controlado por el *view controller* menuvoz

Se ha utilizado UINavigationController para la navegación entre vistas, ya que es lo recomendado y lo que utilizan la mayoría de aplicaciones del mercado, lo que facilita la comunicación entre vistas.

En el momento en el que se detalle cada vista con la ayuda de imágenes y se explique cada una de ellas, resultará más claro.

En el siguiente capítulo se mostrará cada vista y las justificaciones de la realización de estas así como los posibles cambios y mejoras que se podrían llevar a cabo. También es importante detallar en cada parte las constantes para los Autolayout y las funciones

más destacables del código, puesto que se encuentran muchas dificultades en estas cuestiones en el desarrollo de aplicaciones en Xcode.

Estructura del ejercicio

La descripción de cada ejercicio por parte del profesor se debe ajustar al siguiente formato fijo. Esta regularidad permite posteriormente aumentar al máximo la modularidad en el programa, pudiendo cambiar de manera sencilla un ejercicio por otro.

EJERCICIO 1: ELEVACIÓN PELVIS

Acción	Instrucción	Texto-Sintetizador de Voz
1.	1.	Tumbarse apoyando la espalda
	2.	Respire por la nariz
	3.	Piernas flexionadas
	4.	Pies a la anchura de la cadera
	5.	Brazos estirados a los lados del cuerpo

Acción	Instrucción	Texto-Sintetizador de Voz
2.	1.	Inspire
	2.	Despegar la pelvis del suelo hacia el techo
	3.	Suba lentamente vértebra a vértebra
	4.	Expire
	5.	Baje la pelvis en sentido inverso
	6.	Apoye lentamente vértebra a vértebra

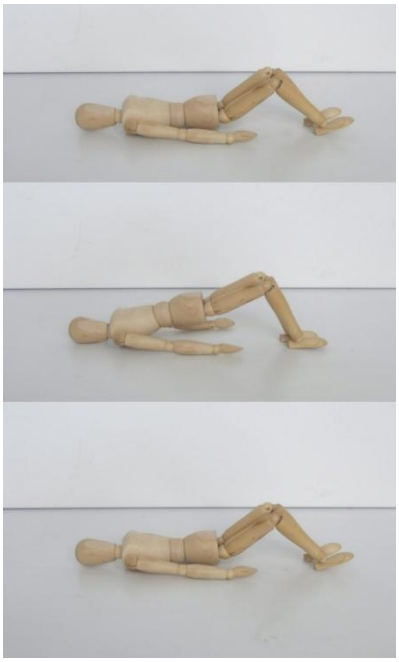


Figura 5 - Desglose de acciones de un ejercicio

Definiendo las instrucciones y acciones se podrá hacer una descomposición perfecta de cada ejercicio. De esta manera, se consigue elaborar una guía sencilla, simplificando al máximo los ejercicios.

En la siguiente imagen se puede apreciar la estructura de todo el proyecto, su *main stroryboard*:

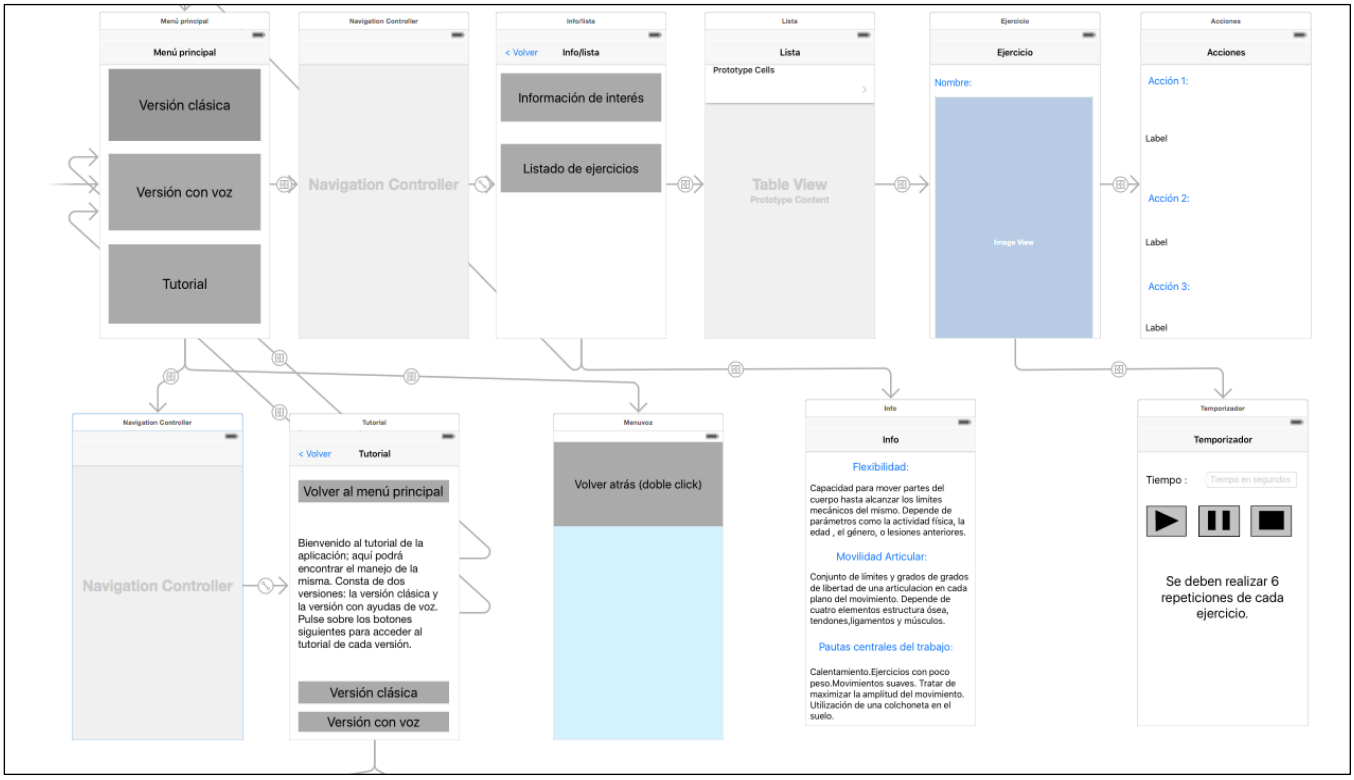


Figura 6 - *Main storyboard*

En esta panorámica se pueden apreciar todas nuestras vistas y ver perfectamente las estructuras y elementos de estas. También se observan las transiciones y conexiones de cada una de ellas aunque se muestra de forma comprimida para que se pueda visualizar todo el proceso en la imagen.

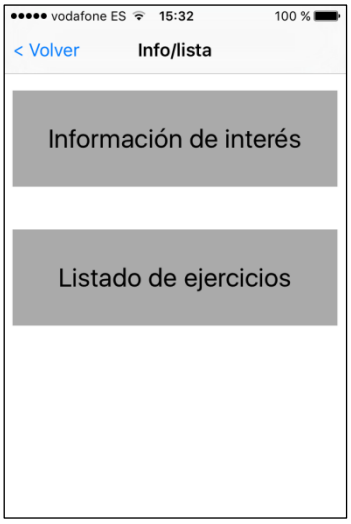
Hay algunas vistas que están controladas por el mismo controlador (viewcontroller), por lo que en cada una de ellas se debe referenciar quién controla los elementos de la vista que deben ser definidos en su cabecera .h y, por supuesto, las vistas que utilicen componentes de otras vistas u objetos, como el caso del sintetizador de voz, deben importar los archivos .h de los objetos o componentes de las vistas utilizados.

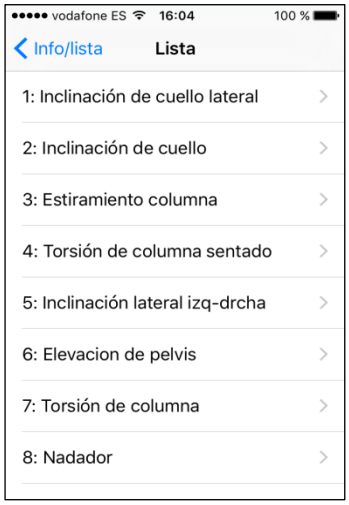
La forma en la cual se crean los viewcontroller y se interconectan es libre y se puede realizar como se quiera; en este caso, las conexiones suelen ser botones o el propio controlador de navegación.

3.4 Comparación con su versión homóloga para Android

Una de las directrices que era preciso seguir era la de seguir la estructura de la aplicación en Android pero adaptada para iOS, lo que se ha tenido siempre en cuenta en este proyecto, manteniendo su tonalidad sencilla así como los menús de esta. Estas imágenes muestran la comparación de dos pantallas de la aplicación:

1) Versión Apple





2) Versión Android

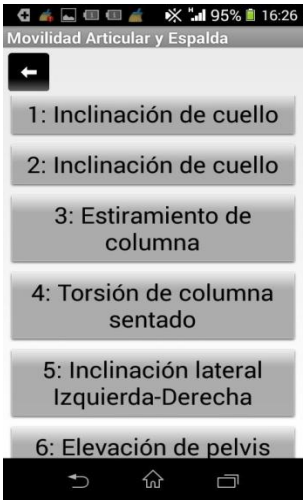


Figura 7 - Imágenes de comparación versión Android y versión Apple

Como se puede comprobar, se parecen aunque sí que se ha echado en falta un degradado del color en el *background* (pantalla de fondo) de los botones que estéticamente resulta más vistoso. Lo único que cambia radicalmente es el menú de voz, que se explicará más adelante.

3. 5 Aplicaciones de estiramientos y yoga en Apple Store

Al buscar aplicaciones sobre estiramientos o yoga se obtienen diversas opciones aplicaciones gratis y de pago. A continuación, se analizan algunas de las aplicaciones descargadas en Apple Store y se comparan con la aplicación desarrollada en el proyecto:

1 Yoga Free

Es una aplicación que contiene una serie de vídeos sucesivos en los cuales se explican diferentes ejercicios y posturas relacionadas con el yoga. En cualquier momento se puede pasar de un vídeo a otro. Es una aplicación muy simple y tiene una interfaz gráfica nula.

La aplicación no posee ninguna opción habilitada para personas con discapacidad, puesto que trata de una serie de vídeos y no podemos utilizar correctamente con Voiceover.

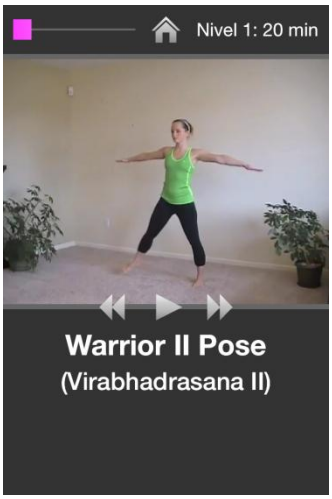


Figura 8 - Imagen de la aplicación Yoga Free

2) Xfit Yoga

La aplicación consta de tres niveles de entrenamiento en los cuales se definen varios ejercicios con un temporizador, imágenes y la definición de las acciones a realizar. Aunque en dicha aplicación las acciones no definen bien el ejercicio, no son exhaustivas por lo que aunque se utilice Voiceover no se entenderían del todo bien los ejercicios.

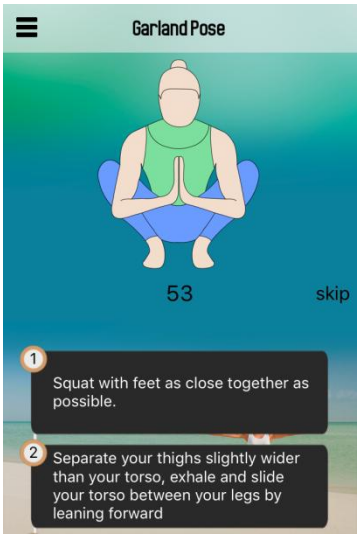


Figura 9 - Imagen de la aplicación Xfit Yoga

3) Estiramientos

Contiene un ejercicio según cada zona del cuerpo y cada uno consta de un temporizador, una imagen y una explicación. Tanto visualmente como por su contenido es una aplicación que se parece bastante a la realizada en el proyecto pero los textos explicativos que contiene no están en formato lectura fácil y al leerlos resultan largos y pesados.

Se podría manejar utilizando Voiceover pero no define del todo bien los ejercicios a realizar.

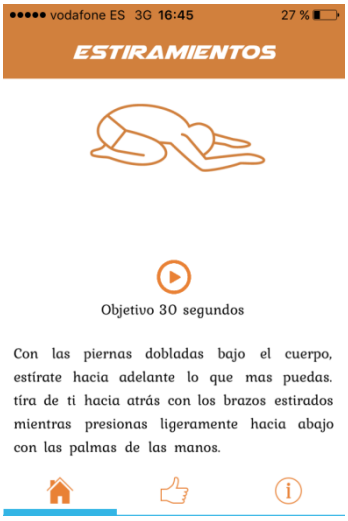


Figura 10 - Imagen de la aplicación Estiramientos

4) Yoga

Consta de un menú con imágenes, textos explicativos y varias informaciones adicionales pero no se puede analizar completamente la aplicación, ya que tiene una versión *premium* de pago. Se podría manejar con Voiceover pero hay muchas opciones y botones en la pantalla y no sería fácil para una invidente poder navegar en esta aplicación ya que los botones son muy pequeños y por ejemplo en un iPhone 4s sería imposible navegar con soltura.

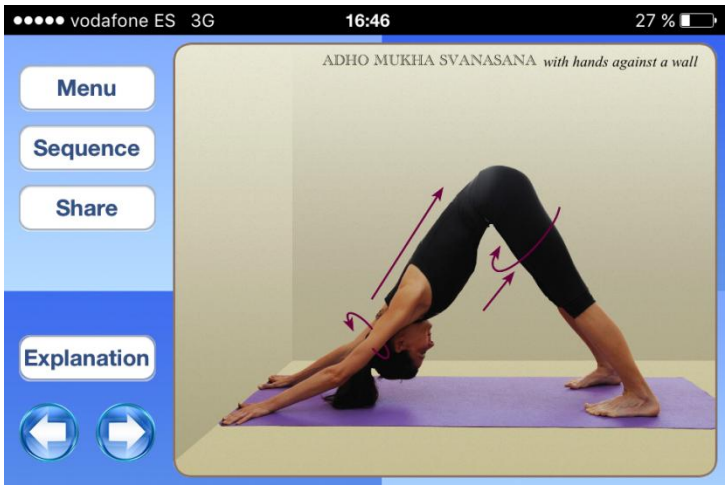


Figura 11 - Imagen aplicacion Yoga

En conclusión, varias de las aplicaciones estudiadas contienen una estructura de los ejercicios como la que tiene la aplicación desarrollada en el proyecto pero ninguna está adaptada a los discapacitados visuales.

Visualmente puede que sean mejores pero los ejercicios que propone la aplicación desarrollada en el proyecto están perfectamente definidos y su adaptación de texto a discurso es perfecta. En el proyecto los ejercicios se centran en la espalda y en la movilidad articular y las aplicaciones estudiadas se centran en rutinas de entrenamiento. También hay que tener en cuenta que algunas de estas aplicaciones son de pago y desarrolladas por un grupo de desarrolladores no solamente por uno.

4. Desarrollo

4.1 Menú principal

Se trata del menú principal en el cual se ve la división del proyecto comentada. A nivel de código no presenta ningún aspecto particular; se utiliza el interface builder y los storyboards. Consta de tres botones que con un segue mostrarán las siguientes vistas.

Este es el resultado en el iPhone 4s:

Menú principal de la aplicación

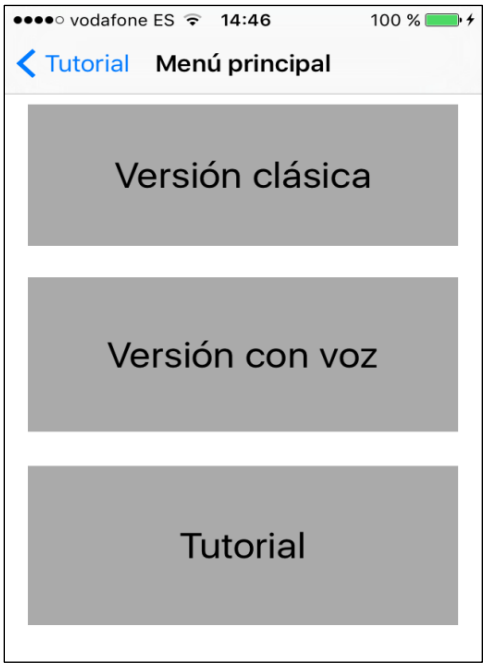


Figura 12 - Imagen menú principal

Las uniones entre las vistas están hechas por segue y su comando show. Con estas uniones y con el UINavigationController se puede navegar entre todas las vistas y ver el título de la vista. Este sistema lo utilizan todas las aplicaciones o bien una barra botones en la parte inferior de la pantalla.

Es el mismo diseño que su aplicación homóloga Android utilizando nuestros botones como herramienta de paso entre vistas, con una interfaz visual seria.

Estas son las opciones para establecer las restricciones en los Autolayout:

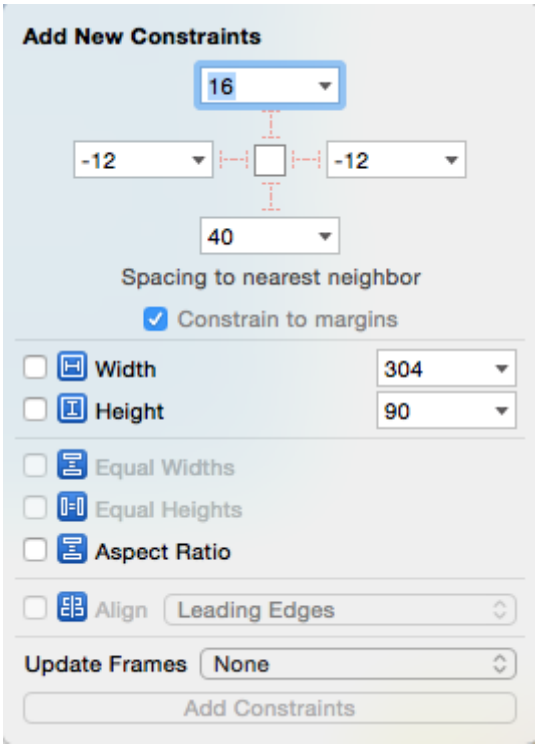


Figura 13 - Opciones del Autolayout

Se puede hacer pin (establecer la distancia) a los límites superior, inferior y laterales del dispositivo, establecer sus dimensiones y posición dentro de la vista, mantener relación de aspecto o mantener el mismo tamaño.

Se puede hacer que dos elementos o más tengan igual altura y anchura, lo que será necesario en algunos menús. Otro elemento importante es definir la distancia entre los elementos, estableciendo espaciado entre los elementos, ya sea vertical u horizontal.

A veces resulta un poco complicado manejar su comportamiento y no es del todo intuitivo. En ocasiones se obtienen diferentes resultados y a la hora de cambiar la vista a modo horizontal o vertical no se obtiene el resultado esperado, lo que hace necesario tener que con configurar todo de nuevo. Cuando se realiza algún cambio pequeño de posición de un elemento, se pueden actualizar las restricciones (*constraints*) y queda actualizada la vista, lo que hace ahorrar mucho tiempo. Por último, se puede hacer que se resuelvan automáticamente los problemas de inconsistencia, pero el resultado la mayoría de las veces no es el deseado.

4.2 Tutorial de la aplicación

El tutorial de la aplicación es la parte más sencilla, ya que no se utiliza código, únicamente el *interface builder* y nuestras marcas del *Autolayout* para su adaptabilidad a todos los dispositivos.

Se han utilizado UIButtons para conectar diferentes vistas mediante el comando *show* (mostrar) así que lo único que hay que hacer es definir y customizar los botones mediante el interfaz.

Este es el resultado en iPhone 4s:

1) Tutorial **2) Tutorial versión clásica** **3) Tutorial versión con voz**

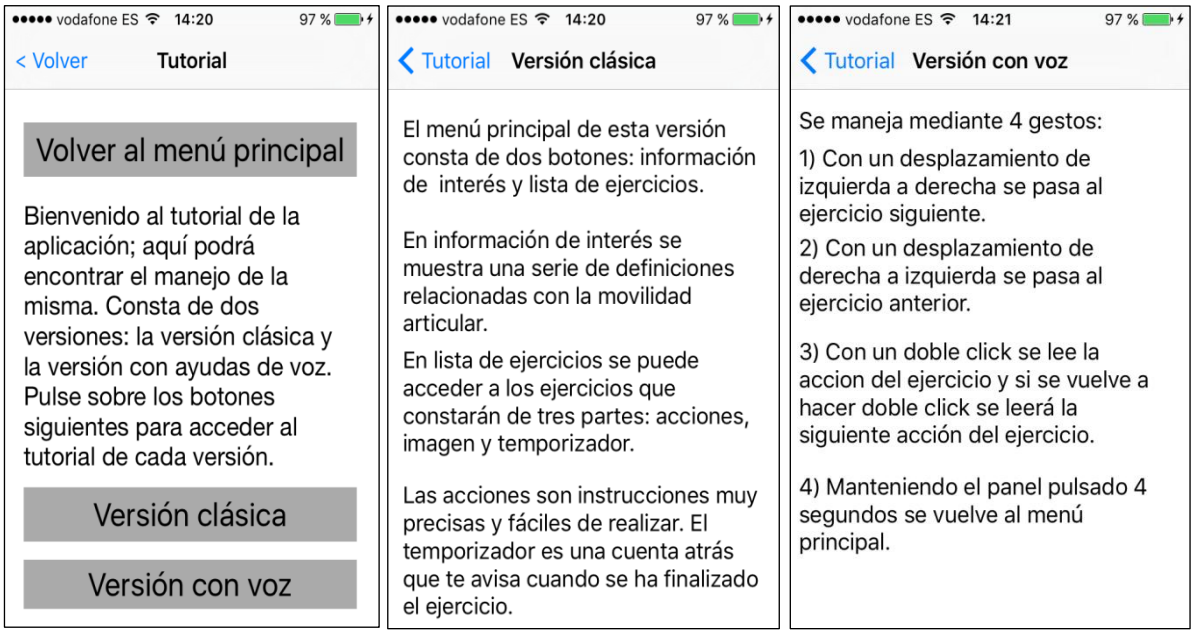


Figura 14 - Imágenes del tutorial

Tras comprobarlo en los tres dispositivos reales iPhone 4s, iPhone5 y iPhone 5s, se ha podido comprobar que se adapta perfectamente. Se trata de menús que son únicamente etiquetas y botones, por lo que no da mucho juego a la hora de realizar una mejora visual. Además, es preciso que sea una aplicación seria de tonos grises, puesto que su predecesora lo era y las imágenes de los ejercicios son de la misma tonalidad.

Se trata de ejercicios fáciles de realizar y con una sencilla implementación, por lo que no tiene sentido hacer un tutorial extenso como el que había en su homóloga para Android.

4.3 Versión clásica

La versión clásica se compone de:

- Información de interés
- Listado de ejercicios. Cada ejercicio consta de :
  - Imagen que explica el ejercicio
  - Acciones
  - Temporizador

En información de interés se definen una serie de conceptos relacionados con la movilidad articular, por lo que sigue siendo una vista sin código que muestra información en unas etiquetas. La novedad reside en la utilización de un *scroller* mediante el *Autolayout.* En esta lista de ejercicios se muestran diez ejercicios y aquí es donde comienza la parte de código llevada a cabo y sus uniones con la interfaz gráfica.

Ahora se mostrará el menú de transición y como queda la vista Info donde se pueden ver las definiciones de información de interés y en la que aparece el nuevo elemento creado, el *scroller*, utilizando UIScrollView.

Menú secundario **Información de interés**

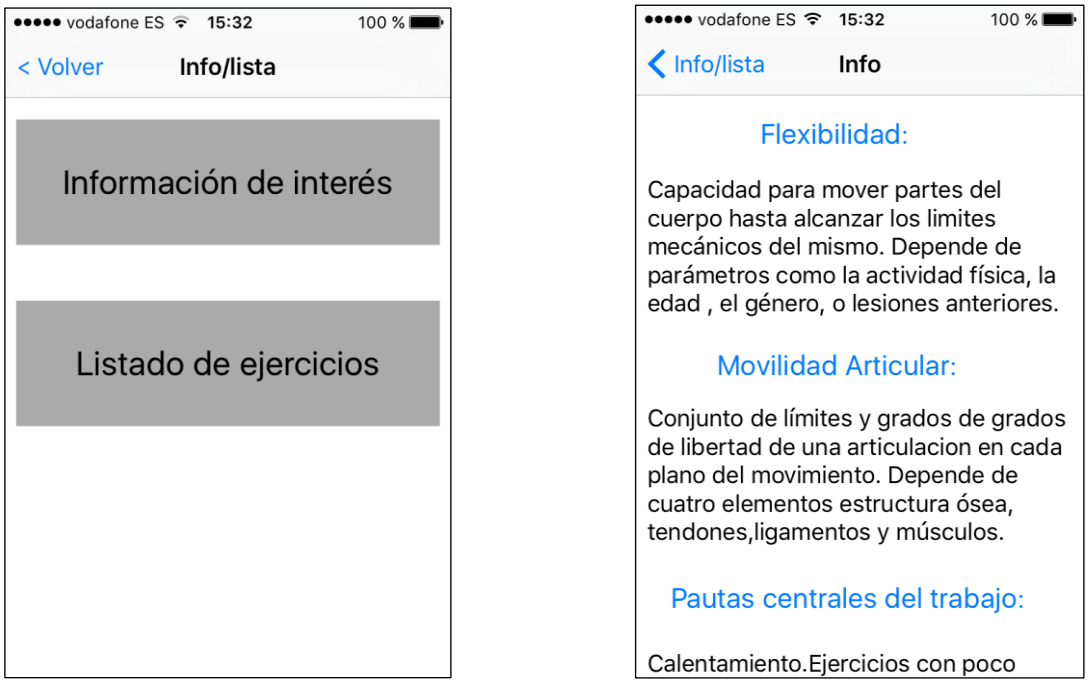


Figura 15 - Imágenes de la Versión clásica

4.3.1 Listado de ejercicios

En listado de ejercicios se ve la lista de los diez ejercicios utilizando un NSObject se crea el modelo del ejercicio que se compone de:

- NSString accion1
- NSString accion2
- NSString accion3
- UIImageView imagen

Una vez creados los ejercicios, se utiliza otro tipo de Viewcontroller, UITableViewController, que es el encargado de manejar la tabla. En el propio código ya se podrán observar las funciones propias que ofrece este nuevo tipo de vista. Para crear la tabla se deberá definir el tamaño de esta y el número de secciones. Existen dos opciones: utilizar un UITableView y buscar las funciones dentro del apartado ayuda del Xcode donde viene las definiciones detalladas de cada elemento o utilizar el UITableViewController. También es preciso configurar las celdas creadas y darles un identificador para poder pasar los datos a la tabla, por lo que las tres funciones que se muestran a continuación son indispensables:

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
    // Return the number of sections.
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
    // Return the number of rows in the section.
    return 10;
}


- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath {
    static NSString *CellIdentifier = @"Cell";
    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier
forIndexPath:indexPath];
    ej = [self.lista objectAtIndex:indexPath.row];
    cell.textLabel.text = ej;
    cell.textLabel.font =[UIFont fontWithName:@"System" size:17];

     return cell;
}
```

Se define el número de secciones, en este caso 1, el número de filas, en este caso 10, también se puede hacer un *count* del *array* ejercicios creado y mediante el

identificador "Cell" pasamos el contenido por fila. Existe la opción de seguir utilizando funciones para configurar la tabla o configurarla mediante nuestras opciones que nos ofrece la interfaz.

Por lo tanto, una vez definidas estas tres funciones y creando dicho *array* de ejercicios, ya se podrá implementar la tabla. El resultado es el siguiente:

Menú Lista

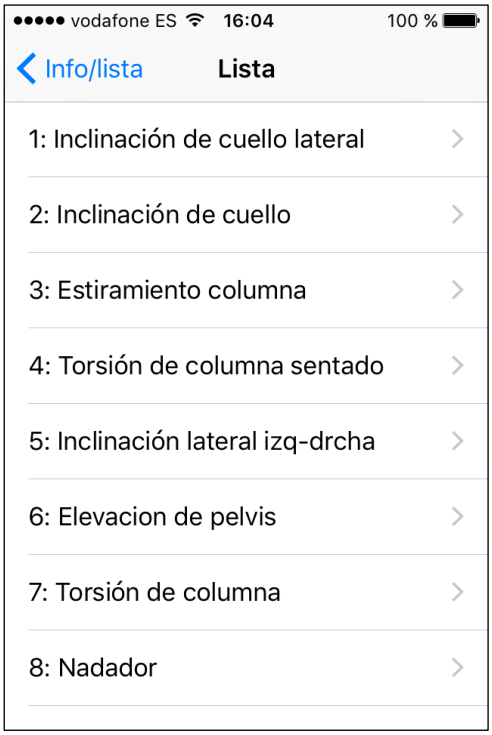


Figura 16 - Imagen lista de ejercicios.

Todavía no está completo porque es necesario que la siguiente vista reciba el ejercicio seleccionado, con lo que se necesitará desarrollar la siguiente función:

```
- (void)prepareForSegue:(UIStoryboardSegue*)segue sender:(id)sender
{
   if ([sender isKindOfClass:[UITableViewCell class]])
   {
      if ([segue.destinationViewController isKindOfClass:[EjercicioViewController class]])
      {
         EjercicioViewController *nextViewController = segue.destinationViewController;
         NSIndexPath *path = [self.tableView indexPathForCell:sender];
         ejelegido = self.lista[path.row];
         nextViewController.ejbueno = ejelegido;
      }
   }
}
```

Se creará un ejercicio en la nueva vista y se igualará al ejercicio enviado y así obtendrá el ejercicio que habíamos seleccionado en nuestra lista de ejercicios. Cada vez que se necesite algo de otra vista o de un objeto, se deberá añadir su cabecera al archivo cabecera .h de dicha vista creada.

La única forma para pasar información entre vistas es mediante este tipo de funciones que se deben implementar. Sería mejor habilitar alguna opción, ya que resulta engorroso tener que añadir código en ambas vistas para realizar este traspaso de datos.

Se pueden crear celdas personalizadas con subtítulos e imágenes y con diferentes tipos de secciones. Estas tablas son utilizadas por la mayoría de aplicaciones como por ejemplo Whatsapp. Lo bueno de utilizar un UITableVIewController es que tanto su Autolayout como conexión con la siguiente vista está ya definido por lo que es mucho más fácil implementar la tabla.

4.3.2 Menú ejercicio

Como se ha comentado anteriormente, cada ejercicio consta de tres partes: una imagen explicativa y muy simple, acciones definidas por el método lectura fácil y un temporizador cuenta atrás para realizar las diferentes series.

Lo primero que hay que hacer es crear un ejercicio e igualarlo al objeto pasado en la anterior vista utilizando la clase creada ejercicio. Como las imágenes son de un formato predefinido, se establece un tamaño grande en mi vista para que se puedan visualizar bien y también los botones para pasar a las acciones y temporizador. En el anterior estaba todo junto pero quedaba un *scroller* muy largo y es preferible utilizar algo que sea divisible y sencillo.

Este es el resultado en el iPhone 4s:

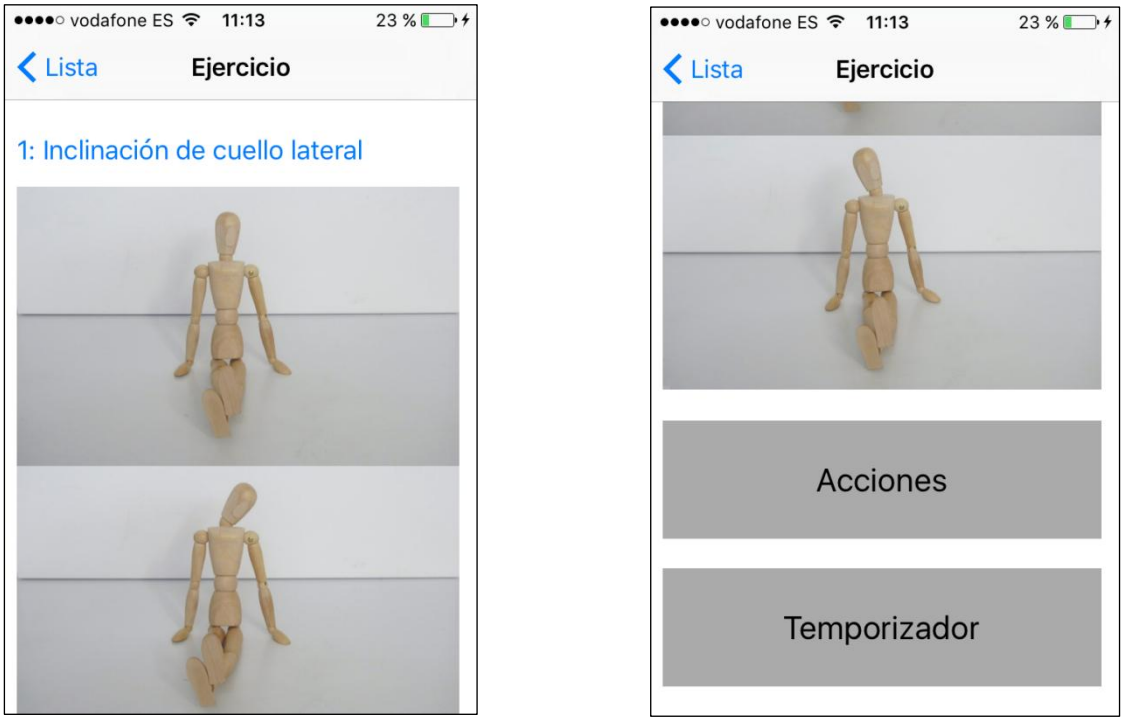


Figura 17 - Imagen menú ejercicio.

Es un menú sencillo basado en tonos grises puesto que la imagen muestra dicho fondo gris. Es preciso destacar que este menú se llevó a cabo de diferentes formas, tanto en código como visualmente, obteniendo el resultado actual.

En el *view controller* acciones se muestran las acciones con explicaciones cortas y dependiendo de la complejidad del ejercicio habrá más acciones según cada

uno de ellos. El temporizador es una cuenta atrás que finalmente da un aviso indicando que se ha finalizado el ejercicio mediante el sintetizador de voz. Para que funcione este menú hay que añadir las cabeceras .h del menú acciones y del menú lista, puesto que se pasan datos al siguiente pero esta vez se pasará una cadena de caracteres, no un índice de un listado como en el caso anterior.

4.3.3 Menú acciones

Para poder reproducir el contenido de las acciones es necesario pasar las cadenas de textos acciones. En la vista Ejercicio es preciso crear un método propio para pasar estas acciones.

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender{
  if ([[segue identifier] isEqualToString:@"Accion1y2"])    {
      Accion1y2 *accion1y2vc = [segue destinationViewController];
       NSString *accion1pasar =_accionetiqueta;
      NSString *accion2pasar =_accionetiqueta2;
      NSString *accion3pasar =_accionetiqueta3;
      NSString *accion4pasar =_accionetiqueta4;
      [accion1y2vc actualizarmisdatos:accion1pasar];
      [accion1y2vc actualizarmisdatos2:accion2pasar];
      [accion1y2vc actualizarmisdatos3:accion3pasar];
      [accion1y2vc actualizarmisdatos4:accion4pasar];
    }
}
```

Se define un identificador para el *segue*, que dirige hacia la otra vista creada, por lo que cada vez que se haga un *show* dándole al botón acciones se pasarán los datos pero la función no está terminada, ya que en la vista Acciones es necesario crear la función, actualizar datos y asignar los cadenas pasadas a cadenas creadas en la propia vista.

```
-(void)actualizarmisdatos:(NSString *)milabel{
   _accion1pasada =milabel;
};

-(void)actualizarmisdatos2:(NSString *)milabel2{
   _accion2pasada =milabel2;
};

-(void)actualizarmisdatos3:(NSString *)milabel3{
   _accion3pasada =milabel3;
};

-(void)actualizarmisdatos4:(NSString *)milabel4{
   _accion4pasada =milabel4;
};

self.explicacion1.text = _accion1pasada;
self.explicacion2.text = _accion2pasada;
self.explicacion3.text = _accion3pasada;
self.explicacion4.text = _accion4pasada;
```

De este modo, utilizando un *scroller* y los elementos seleccionados se crea una vista con etiquetas (UILabel) que mostrarán el contenido de las acciones.

El resultado en iPhone 4s es el siguiente:

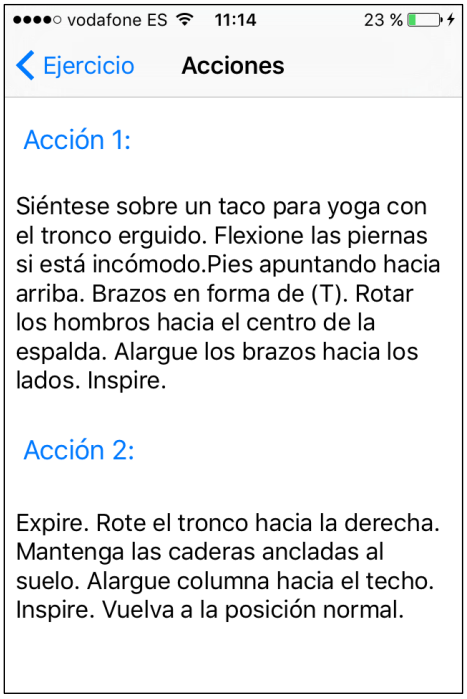


Figura 18 - Imagen menú acciones.

4.3.4 Menú Temporizador

En este menú se representa una cuenta atrás de un minuto de duración o menos según establezca el usuario el tiempo que quiera estirar y unos botones personalizados de Play, Pause y Stop. Al presionar el campo texto, aparece un panel numérico donde se introduce el tiempo y dando al botón Play se inicia la cuenta atrás. Al finalizar el sintetizador de voz, este avisará de su finalización. En todo momento se puede parar la cuenta o establecer otro valor pulsando el botón Stop.

Figura 19 - Imagen del temporizador.

A nivel de de código se debe establecer un *timer* (NStimer), crear una función para Start, Pausa y Stop y por último utilizar el sintetizador de voz. En siguientes contenidos se explicará con detalle el sintetizador de voz.

De este modo, se puede separar esta parte y utilizar el temporizador en otra aplicación. También es necesario añadir la cabecera del sintetizador y crear el objeto sintetizador en la vista para poder utilizar esta función. Se podría haber utilizado un tono o algún elemento parecido, pero el sintetizador se consideró la opción más satisfactoria.

```
- (IBAction)start:(id)sender {
    contador = [_segundostext.text intValue];
    NSLog (@"el numero es %i",contador);
    self.start.hidden = YES;
    [self.segundostext resignFirstResponder];
        self.start.hidden = NO;
        contador = 0;
    }
    timer = [NSTimer scheduledTimerWithTimeInterval:1 target:self selector:@selector(count) userInfo:nil repeats:true];
}
-(void)count{
    if(contador == 0)
    {
        _segundostext.text = @"0";
        [timer invalidate ];
    }
    else {
        contador = contador - 1;
        _segundostext.text = [NSString stringWithFormat:@"%i",contador];
        if (contador == 0){
            [timer invalidate];
            if (sintetizador == nil)
            {
              sintetizador = [[SintetizadordeVoz alloc] init];
            }
            [sintetizador textoavoz:@"Enhorabuena, ejercicio completado"];
            self.start.hidden = NO;
        }
    }
}
```

Una de las cosas más favorables es el autocompletar que tiene Xcode, ya que cuando se está implementando un método este ayuda bastante, aunque en función del objeto que se utiliza las opciones son muy diferentes y en ocasiones en difícil encontrar lo que se está buscando. Un ejemplo de ello es el hecho hacer algo tan sencillo como que el teclado numérico desaparezca, lo que puede resultar muy engorroso.

4.4 Menú de voz

4.4.1 Sintetizador de voz

Se ha elegido el sintetizador de voz AVSpeechSynthesyzer, ya que lo que interesaba era tener un sintetizador de voz de texto a discurso, no utilizar herramientas como Siri o Voiceover, que ya tienen sus propios métodos por lo que mediante el reconocedor de gestos y el sintetizador de voz se puede realizar un menú que dé la información suficiente para que una persona con discapacidad pueda realizar los ejercicios mostrados en la aplicación.

A nivel de código se creó un NSobject, un objeto para poder llamar todas las veces que desee a nuestro sintetizador. En este objeto se llama a AVSpeech mediante la librería AVFoundation.h y así se consigue acceder a todos sus elementos y poder configurar su velocidad, pronunciación y todas sus características.

```
- (void)textoavoz:(NSString *)habla {

   if (sintetizador == nil) {
      sintetizador = [[AVSpeechSynthesizer alloc] init];
      [sintetizador setDelegate:self];
   }
   NSArray* discurso= [habla componentsSeparatedByString:@"|"];
   [self setMyExpectedUtterances:[discurso count]];
   for (int i=0; i<[discurso count]; i++)
   {
      NSString* speech = [discurso objectAtIndex:i];
      AVSpeechUtterance *utterance = [AVSpeechUtterance speechUtteranceWithString:speech];
      [utterance setVolume:1.0];
      [utterance setRate:0.375];
      [utterance setPostUtteranceDelay:0.15];
      [self setMyDeactivationAttempts:0];
      [sintetizador speakUtterance:utterance];
   }
 }
```

Se intentó de varias formas realizar el código para llamar al sintetizador, pero al final la que dio mejor resultado era la de crear un objeto propio y definir la *speakUterrance*, la cual permite modificar los parámetros de la lectura. La calidad de voz que tiene este sintetizador supera con creces el utilizado en la aplicación homóloga para Android, ya que el sintetizador es mucho mejor. Al hacerlo así este código servirá para poder utilizar el sintetizador en cualquier aplicación siempre que se quiera.

4.4.2 Menú de voz

Al principio se planteó este menú como lo hizo la versión de la aplicación realizada para Android pero no era la mejor opción, puesto que su manejo era muy poco intuitivo y para una persona con discapacidad no servía de ayuda. La primera versión que se realizó, muy similar a la de su homóloga Android, era la siguiente:

Figura 20 - Versión alternativa menú de voz.

El funcionamiento es el siguiente: si se mantiene pulsado se lee el título del botón y si se hace doble clic, dependiendo del botón pulsado, se lee el contenido del botón o se pasa a otra vista. Por ejemplo, al pulsar el botón flexibilidad se lee la palabra flexibilidad y si se hace doble clic se lee la definición de flexibilidad. Si se desplaza el dedo de izquierda a derecha en el último menú en la zona blanca, se pasa al ejercicio siguiente y desplazándolo de derecha a izquierda se vuelve al ejercicio anterior.

En conclusión, el objetivo de este proyecto era transformar la aplicación y por ello se decidió cambiar esta versión de ayuda con voz, para hacer que este menú fuera más manejable para una persona con discapacidad.

Por lo tanto, en vez de tener tantos menús, se llegó a la conclusión junto con el tutor de este proyecto, de que con un solo menú que contuviese la información de los ejercicios era suficiente y un manejo mucho más sencillo, ya que cambió por completo su funcionabilidad y se hizo de este modo más accesible para una persona con discapacidad. Como la aplicación tiene dos versiones dentro de la misma, no es necesario utilizar Voiceover o Siri ni ningún método ya implementado.

Con cuatro gestos se puede realizar un manejo adecuado, así como moverse entre los ejercicios y las acciones que componen los mismos. Los gestos son los siguientes:

- **Desplazamiento del dedo de izquierda a derecha:** se pasa al ejercicio siguiente.
- **Desplazamiento del dedo de derecha a izquierda:** se pasa al ejercicio anterior.
- **Doble clic:** se lee la acción 1 del ejercicio donde se encuentre en ese momento y si se vuelve a pulsar se lee la siguiente así hasta volver otra vez a la acción 1.
- **Si se mantiene pulsado cuatro segundos,** se vuelve al inicio de la aplicación.

Si por equivocación se entra a esta versión en la parte superior de la pantalla haciendo doble clic, se vuelve al menú principal. Al entrar en la versión con ayuda de voz, se indica el primer ejercicio y cada vez que se pasa de un ejercicio a otro se indica en todo momento en qué ejercicio se encuentra uno, haciendo que prácticamente con dos gestos, desplazar lateralmente y hacer doble clic, se pueda acceder a toda la información, ya que mediante el método que se explicó anteriormente se pueden sintetizar todas las acciones de dicho ejercicio y se puede pasar de texto a sintetizador fácilmente.

A nivel de código hay que establecer las funciones para pasar de un ejercicio a otro, el método de leer las instrucciones, establecer los gestos y una función para poder salir del ejercicio. Para los gestos se utilizará UIGestureRecognizer y se establecerá su acción dependiendo del gesto realizado.

Se muestra un ejemplo de un gesto y la función en la que se vuelve al menú principal:

```
- (IBAction)ejsig:(UISwipeGestureRecognizer *)sender {
   if ( contador == 9){
      contador = 9;
   }
   else{
      contador = contador + 1;
   }
   Ejercicio *ejseleccionado1 = self->ejercicios[contador];
   if (sintetizador == nil)
   {
      sintetizador = [[SintetizadordeVoz  alloc] init];
   }
   [sintetizador textoavoz:ejseleccionado1.nombre];
   contadoraccion = 1;
   }
```

```
- (IBAction)salir:(UILongPressGestureRecognizer *)sender
{
   ViewController *vc1 = [self.storyboard instantiateViewControllerWithIdentifier:@"vc1"];
   [self presentViewController:vc1 animated:NO completion:nil];

}
```

El resultado final es el siguiente:

Figura 21 - Imagen menú de voz.

El panel que utiliza la persona con discapacidad es la zona celeste y en toda esa zona reconocerá los cuatro gestos ya comentados. Por ello, se ha evitado el uso de botones y se ha simplificado al máximo para que el sintetizador y el panel creado sean suficientes para navegar entre los ejercicios y las acciones que componen estos. Por supuesto, el menú se adapta a todos los dispositivos y el sintetizador de voz funciona igual en las pruebas realizadas en el simulador, tanto en el iPhone 4s como en el iPhone 5.

4.5 Launch screen.xib, UIAlertViewController y UIScrollView

4.5.1 Launch screen.xib

Se trata de un nuevo elemento introducido en iOS 8. En el tiempo en el que carga la aplicación aparece una pantalla de lanzamiento que puede ser por ejemplo un logo o una imagen o lo que se crea oportuno, por ejemplo en la aplicación de Youtube aparece el logo de Youtube. En la pantalla de carga aparece un elemento que contienen todas las aplicaciones realizadas por el DSLab de la escuela que muestra el logo y la información sobre quién ha realizado la aplicación.

Figura 22 - Pantalla de carga de la aplicación

4.5.2 UIAlertViewController

Cuando se instaló la nueva del iOS (iOS 9) apareció un *warning* avisando que el tipo de alerta utilizado en el proyecto iba a ser eliminado en futuras versiones de iOS. La alerta se utiliza para notificar que el número introducido en el temporizador era muy alto y que el estiramiento debía ser más corto.

Se podría haber optado por no cambiarlo pero se decidió modificar la alerta y utilizar UIAlertViewController. De este modo, aunque el resultado es el mismo, la forma de implementar cambia. Antes se implementaba así:

```
UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"Error" message:@"Introduce un tiempo menor o igual a 60 segundos" delegate:nil cancelButtonTitle:@"Ok" otherButtonTitles: nil];
[_segundostext resignFirstResponder ];
[alert show];
```

Ahora se implementa de este modo:

```
UIAlertController * alert=   [UIAlertController
                              alertControllerWithTitle:@"Error"
                              message:@"Introduce un tiempo menor o igual a 60 segundos"
                              preferredStyle:UIAlertControllerStyleAlert];
    [self presentViewController:alert animated:YES completion:nil];

UIAlertAction* ok = [UIAlertAction  actionWithTitle:@"OK"
          style:UIAlertActionStyleDefault    handler:^(UIAlertAction * action)
                    {  [alert dismissViewControllerAnimated:YES completion:nil];
                    }];
[alert addAction:ok];
```

4.5.3 UIScrollView

Hubo algunos problemas al implementar un *scroll* porque lo que se intentaba era hacer el *scroller* mediante código en vez de usar el Autolayout y el *interface builder* creando un UIScrollView, habilitando su propiedad de *scroll* y dándole un tamaño mediante CGPointMaker y funcionaba, pero no se podían ajustar los elementos a los diferentes tamaños de otros modelos iPhone, por lo que el resultado no era del todo satisfactorio.

La segunda opción fue no utilizar un *scroller* pero el tamaño del iPhone 4s no es muy grande y por lo tanto algunas vistas quedaban muy pequeñas. Además, la foto del ejercicio no mantenía la relación de aspecto y el resultado no era el esperado.

Como resultado, se implementó el *scroller* mediante Autolayout, aunque se seguía sin poder adaptar el contenido dentro del *scroller*.

Finalmente, se encontró una solución. No obstante, dicho procedimiento no resultó nada fácil, con lo que se considera que algo tan básico debería venir explicado de forma más clara y sencilla.

Cada vista tiene un UIView principal que contiene todos los elementos de la vista botones, etiquetas, campos de texto o lo que establezcamos. Hay que poner un *scroller* que ocupe las dimensiones de la vista principal y crear una vista dentro del *scroller* que sea la vista contenido que tenga las dimensiones de altura que se desee para el *scroller* y que sea igual de ancho que el *scroller* creado.

Vista principal --> Scroller --> Vista contenido

La solución consta de cuatro etapas:

1) Se debe hacer un *pin* al *top*, un *pin* a los laterales y uno al *bottom* con el *scroller* de modo que cada vez que cambie de dispositivo o de tamaño quedará ajustado a los límites.

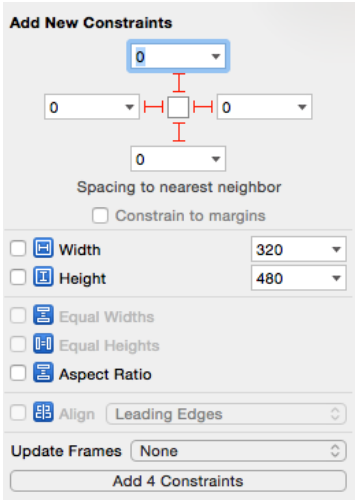


Figura 23 - Constraints del scroller.

2) En la vista contenido se debe hacer un *pin* a los cuatro ejes y establecer fija su altura que es la que definirá la altura del *scroller* creado.

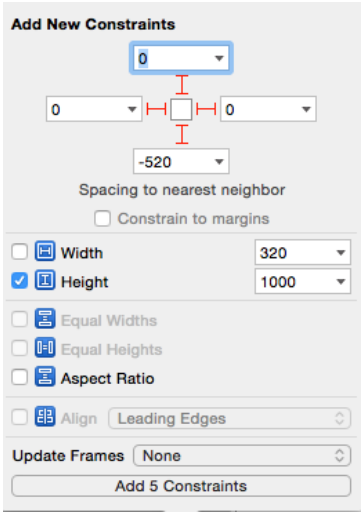


Figura 24 - Constraints de la vista contenido.

3) Hay que mantener pulsada la tecla cmd y clicar al *scroller* y a la vista contenido y establecer la misma anchura, por lo que todas nuestras restricciones se marcan en azul y podremos tener el *scroller* listo sin utilizar código.

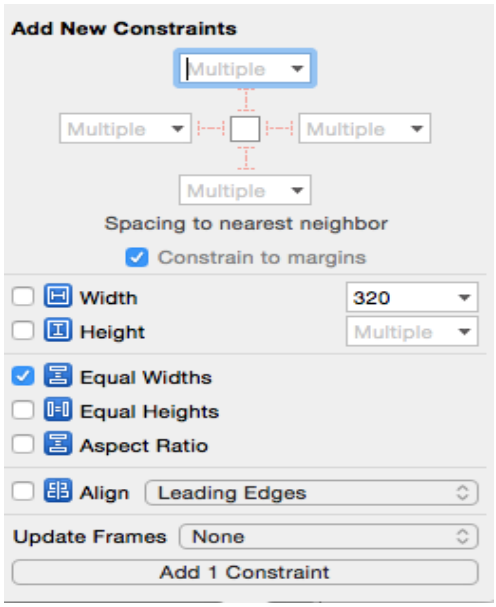


Figura 25 - Igualar anchura del scroller a la anchura de la vista contenido.

4) Por último, es preciso cambiar la constante de la vista contenido y establecerla a 0 para ir al inicio del *scroller.*

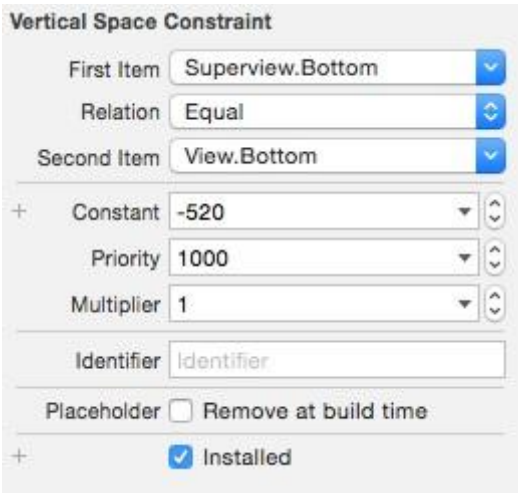


Figura 26 - Cambiar constante vertical a 0.

Como se ha podido observar, se trata de un proceso poco intuitivo y bastante difícil de llevar a cabo, pero juntando varias informaciones se consiguió que funcionara favorablemente.

5 Pruebas y resultados:

Al empezar a programar en el Xcode se pudo apreciar que el compilador era muy lento y que el ordenador no funcionaba correctamente. El iOS Simulator tardaba cinco minutos en mostrar el resultado si es que lo mostraba e incluso se pensó en abandonar el proyecto puesto que no era viable. En iOS 8 para poder probar una aplicación en un dispositivo físico se necesita inscribirlo en el programa de desarrolladores. Una vez inscrito un iPhone 4s se realizó la correspondiente prueba compilando y ejecutando una aplicación y no hubo ningún problema, por lo que el método de trabajo fue siempre probar las aplicaciones en el iPhone 4s y así se decidió continuar con dicho proyecto.

Como muestra la imagen, al crear las vistas, siempre se puede previsualizar un resultado previo en los diferentes terminales:

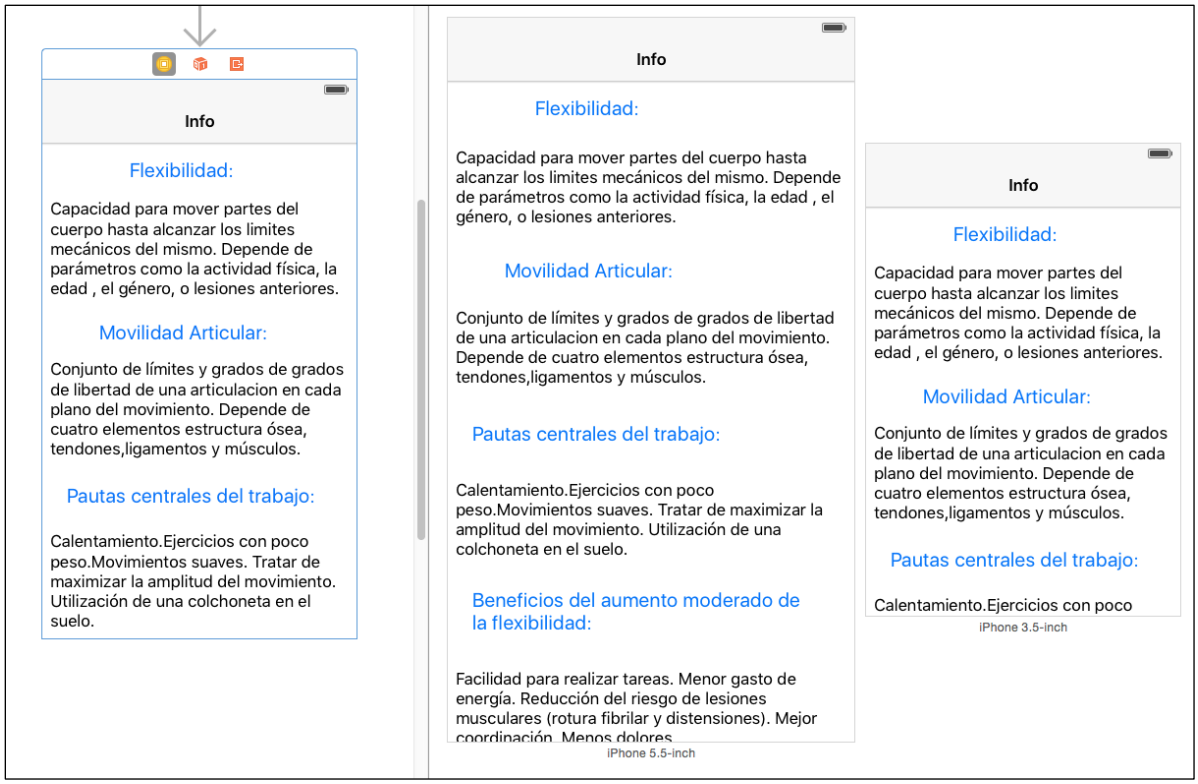


Figura 27 - Previsualización de una vista.

Por esta razón, siempre se ha tenido un control exacto del resultado en otros modelos de iPhone. Se puede utilizar el iOS Simulator para ver el resultado aunque tarda alrededor de cinco minutos o más. Por ello, solo se comprobó una vez finalizada la programación de la aplicación para de este modo visualizar el resultado final. Con un ordenador más potente estos problemas probablemente no hubieran surgido.

La aplicación fue probada y testeada en su versión homóloga Android, por lo que a las pruebas que se hicieron se añaden las realizadas en esta versión. En cuanto a rendimiento, se trata de una aplicación sencilla que no consume apenas recursos del procesador, tampoco consume datos y funciona perfectamente, por lo que no resulta relevante explicar este aspecto.

Siempre se han realizado las pruebas en un iPhone 4s, que es el modelo existente con menos potencia y características inferiores en el que se puede programar aplicaciones para iOS 9, por lo que en el resto de modelos funciona sin ningún problema y, en efecto, al probarlo en un iPhone 5 funciona perfectamente.

En cuanto a la versión de ayuda con voz, se navega sin dificultad en el panel táctil que reconoce los gestos y no tiene ninguna complicación, ya que el guiado de voz es exhaustivo e informa sobre en qué ejercicio uno se encuentra en todo momento. Se ha realizado una prueba con una persona con discapacidad y lo cierto es que no ha tenido dificultades a la hora de navegar en este menú.

Asimismo, hay que tener en cuenta que en su versión homóloga para Android se hicieron varias pruebas y funcionaba a pesar de que tenía un menú más complejo, por lo que en esta versión tampoco debería haber ningún problema.

También se han realizado pruebas sobre el iOS Simulator para un iPhone 6 y se adapta perfectamente ya que los Autolayouts están perfectamente diseñados y todas las restricciones (*constraints*) están bien definidas y no se observan errores ni avisos al compilarlo.

6. Conclusiones, trabajos futuros y aprendizaje:

6.1 Conclusiones

A grandes rasgos, tengo que decir que ha sido una experiencia muy enriquecedora, ya que me ha permitido utilizar por primera vez un ordenador MAC y aprender acerca de este sistema operativo y de su funcionamiento. No obstante, en mi opinión, los requisitos que pide Apple me parecen un poco altos, puesto que es preciso disponer de un ordenador, un móvil o tableta y pagar, por supuesto, una licencia de administrador.

Por otra parte, el compilador Xcode y toda su interfaz gráfica son realmente buenos y ofrecen muchísimas posibilidades, ya que en todo momento se pueden probar los resultados en tu dispositivo o en el simulador, puesto que con iOS 9 han hecho que esto sea posible sin necesidad de tener una licencia.

Sin embargo, puesto que el número de programadores no es muy elevado, resulta muy complicado encontrar información y códigos de ejemplo. Xcode y su programación en Objective-C a veces muestran cómo programar y hacer los pasos siguiendo su modelo, lo que hace que resulte mucho más costoso avanzar, aunque sus estándares estéticos son mayores que los de Android.

En definitiva, si te gusta la marca Apple es una gran experiencia y oportunidad aprender a programar y desarrollar aplicaciones, ya que una vez aprendidos los métodos estos son fáciles de aplicar. No obstante, pienso que hoy en día es muchísimo más fácil aprender a desarrollar aplicaciones para Android y se aprende más rápido.

En cuanto a la aplicación que se ha llevado a cabo en este proyecto, considero que es una forma sencilla de realizar una serie de estiramientos. Además, es apto para todo tipo de público, así como para aquellas personas con problemas de visibilidad o con algún tipo de discapacidad, puesto que el guiado de voz les permite realizar dichos estiramientos sin ningún problema. Es cierto que hay muchas más aplicaciones de yoga y de estiramiento, pero considero que la forma de explicar los ejercicios y las fotos utilizando maniquís son mucho más descriptivas y profesionales, puesto que toda la aplicación se hace con material propio y no la típica foto de una chica haciendo una postura sobre fondos coloridos en la que generalmente no se aprecia bien cómo hacer el ejercicio.

En conclusión, mi aplicación aporta algo diferente respecto al resto de aplicaciones existentes en el mercado, razón por la que tanto mi tutor como yo decidimos que esta sería la más adecuada e interesante.

6.2 Aprendizaje

Durante cada una de las etapas de desarrollo de esta memoria, se pudieron adquirir diversos conocimientos:

Aprender el funcionamiento de OS X Yosemite:

Al principio, se tuvo que aprender a interactuar con el sistema operativo de Apple y actualizar sus programas, puesto que estaban desfasados. Se tuvo que actualizar el sistema operativo al OS X Yosemite para que pudiese correr la nueva versión del Xcode 6.0 y actualizar la antigua versión de Xcode que tenía el ordenador. Una vez concluido este paso, el siguiente paso fue enrolarse en el grupo de desarrolladores de Apple de la Universidad Autónoma de Madrid y aprender a utilizar el sistema de llaveros de Apple para los certificados.

Aprender a utilizar Xcode y su lenguaje Objective-C:

Al comenzar a investigar el mundo de Xcode, aparecieron dos posibilidades: programar en Objective-C o en Swift. Este último acaba de salir al mercado y no presentaba mucha información ni ejemplos, por lo que se escogió Objective-C, aunque ambos lenguajes son muy similares.

Se empezó el proceso de aprendizaje de Objective-C mediante un curso de iOS 8 en bitfountain.io, curso en inglés con más de 20 temas y 20 aplicaciones que sirvió como introducción en el mundo de la programación en Xcode y Objective-C y también se consultó durante el proceso varios libros de la biblioteca de la escuela.

Aprender adaptar el diseño utilizando Xcode:

La base del proyecto era adaptar la aplicación homóloga para Android, por lo que había que realizar un diseño parecido utilizando los elementos que nos ofrecen el UIKit en Xcode.

Aprender a verificar y a realizar pruebas:

Durante el proceso de la aplicación siempre se realizaron pruebas en el iPhone 4s y se hicieron varias consultas al foro de desarrolladores de Apple y también se utilizó la ayuda del propio Xcode, en la cual viene la referencia de todas las clases y sus diferentes métodos y formas de implementarlos. Para cualquier clase que se vaya a utilizar es imprescindible consultar esta ayuda puesto que informa sobre todas las propiedades y funciones de cada clase.

Aprender el proceso de certificación:

Uno de los últimos pasos consistió en aprender el largo proceso de certificación de la aplicación para Apple Store.

Aprendizaje continuo a lo largo de la realización:

Debido a los diversos y continuos cambios (se empezó en el iOS7 pasando por iOS 8 y por el actual iOS 9), se tuvieron que cambiar partes del código porque iban a ser eliminadas o cambiaban la forma de implementarlas. Todos estos cambios ocurrieron en menos de un año.

6.3 Trabajos futuros

En cuanto a las posibles mejoras o trabajos futuros de esta aplicación, una opción muy recomendable sería hacer una versión diferente para los iPads, ya que el tamaño de estos dispositivos es muy grande y con la forma en que está hecha la aplicación no se obtendría el mejor resultado puesto que se mostrarían botones gigantes y letras muy grandes, razón por la que se ha optado por realizar la aplicación para iPhone.

Otro posible cambio sería añadir varios idiomas puesto que en el sintetizador de voz se puede configurar el idioma pero habría que traducir todos los textos y menús.

Por otro lado, se podrían añadir más estiramientos, ya que existen muchos más estiramientos. Asimismo, se podría diseñar la aplicación de otro modo, puesto que el proyecto exigía hacer una aplicación parecida a su homóloga para Android. Se podría modificar tanto los tonos grises que dan un aspecto muy serio a esta, como el formato, que también podría modificarse.

A nivel personal, como soy un gran amante del deporte, me gustaría realizar una aplicación de calidad en este sector. En mi opinión, esto junto con el hecho de que no he visto una aplicación bien realizada que explique bien las rutinas y los ejercicios que se deben hacer, me motiva mucho a la hora de realizar un proyecto futuro que abarcase este tema de forma clara y completa. Se trataría sin duda de un trabajo muy enriquecedor a nivel tanto personal como profesional.

En definitiva, todas las aplicaciones deberían tener una versión con ayudas de voz o estar adaptadas para ser utilizadas con Voiceover o cualquier tipo de sistema de guía por voz, por lo que muchos proyectos ya existentes o aplicaciones se podrían adaptar y ser mejoradas para llegar a todos los usuarios.

7. Referencias:

- A.Lopez Hernandez, Fernando. "Objectice C curso práctico para programadores, MAC , OSX, iPhone y iPad", Editorial Grupo RC.
- Stephen G. Kochan , "Programación con Objective - C", Editorial Anaya.
- Página de desarrolladores de Apple https://developer.apple.com/
- Curso en bitfountain.io para aplicaciones en iOS 7 (Objective - C) curso en ingles dirigido por Eliot Arntz. https://www.bitfountain.io
- http://www.unavarra.es/digitalAssets/173/173488_100000Subir-una-aplicacion-a-la-App-Store.pdf publicado por la Universidad Pública de Navarra UPNA
- Aplicación homóloga Android https://play.google.com/store/apps/details?id=com.movilidadarticularyespalda&hl=es
- http://www.medicalnewstoday.com/articles/206576.php, artículo publicado por Dr. Zrenner de Medical News Today
- http://www.uncu.org.uy/tecnologia_y_ceguera.htm Unión Nacional de ciegos del Uruguay
- http://www.abc.es/salud/noticias/20140702/abci-cornea-regenerar-nature-celula-201407021706.html artículo publicado por ABC
- http://www.apple.com/es/accessibility/ios/ Página de Apple
- http://www.ceditec.etsit.upm.es/index.php?option=com_content&view=article&id=22087%3Aaplicaciones-moviles-para-personas-con-discapacidad-visual&catid=40&Itemid=1440&lang=es publicado por el Centro de Difusión de Tecnologías (CEDTEC).
- http://www.abc.es/tecnologia/moviles-telefonia/20140725/abci-ventas-smartphones-tableta-espana-201407251123.html publicado por ABC
- http://www.sportfactor.es/blog/2012/03/la-importancia-de-los-estiramientos-en-el-deporte/
- http://www.applesfera.com/ios/la-cuota-de-mercado-de-ios-sigue-subiendo-lentamente-en-europa-segun-cifras-de-kantar
- http://www.efesalud.com/noticias/el-54-de-los-espanoles-nunca-o-rara-vez-hace-deporte/

8. Equipamiento:

Mac mini 5.1:

Información del hardware:	
Nombre del modelo:	Mac mini
Identificador del modelo:	Macmini5,1
Nombre del procesador:	Intel Core i5
Velocidad del procesador:	2,3 GHz
Cantidad de procesadores:	1
Cantidad total de núcleos:	2
Caché de nivel 2 (por núcleo):	256 KB
Caché de nivel 3:	3 MB
Memoria:	2 GB
Versión de la ROM de arranque:	MM51.0077.B12
Versión SMC (sistema):	1.76f0
Número de serie (sistema):	C07GPBHSDJD0
UUID de hardware:	89909657-B3E9-545D-A34F-B109C5E90505

Tabla 3 - Mac mini especificaciones técnicas

Figura 28 - Mac mini

Móvil iPhone 4s:

Medidas	**58.60 mm ancho x 9.30 mm grosor x 115.20 mm alto**
Brillo	**250 cd/m²**
Contraste	**Estático 1000:1 Dinámico 10000000:1**
Batería	**1432 mAh puede durar hasta 14 horas**
Memoria	**512 mb RAM y 16GB de disco duro**
Tamaño	**3.5 pulgadas**
Tecnología de imagen	**IPS**
Conectividad	**Wifi, USB 2.0, redes 2G y redes 3G**
Densidad por pixeles	**330 ppp** Cámara: **8 megapíxeles**

Tabla 4 - Especificaciones técnicas iPhone 4s

Figura 29 - iPhone 4s.

Pantalla HP Pavilion 22xi:

Ángulo de visualización **178° (H) 178° (V)**
Brillo **250 cd/m²**
Contraste **Estático 1000:1 Dinámico 10000000:1**
Formato **Panorámica (16:9)**
Resolución **1920 x 1080**
Tamaño **21.5 pulgadas (54.6 cm)**
Tecnología de imagen **IPS**
Tecnología 3D **No**
Calidad de imagen **Full HD**

Tabla 5 - Especificaciones técnicas iPhone 4s

Figura 30 - Pantalla HP Pavilion.

Teclado Apple MB110:

Figura 31 - Teclado Apple.

9. Anexos

Anexo 1)

Recomendaciones de Lectura Fácil para programadores

Cada ejercicio debe desglosado en una serie de indicaciones sencillas, basadas en el estilo "Lectura Fácil" y apropiadas para realizar comandos por Sintetizador de Voz en un teléfono inteligente.

Las principales opciones de lectura fácil aplicables a apps son:

- El punto es el signo ortográfico fundamental para la separación de contenidos.
- Se evita el punto y seguido.
- El punto y aparte se usa para separar párrafos e ideas.
- No hay coma, punto y coma ó puntos suspensivos.
- No hay corchetes y signos ortográficos poco habituales
- Los números en cifras, no con letras.
- No hay numeración romana.
- Se evitan casi todos los tiempos y modos verbales: futuro, subjuntivo, condicional, formas compuestas.
- Se evitar la omisión del sujeto.
- Se utilizan oraciones simples cortas, con la estructura «sujeto + verbo+ complementos».
- Se evitan oraciones complejas.
- Se escribe de modo concreto, simple y directo.
- Cada línea contiene una idea (en la app, una posición o movimiento).
- Si la información es extensa, se corta en frases separadas.
- Se elimina todo contenido, ideas, vocablos y oraciones innecesarias.
- Se trata de no dar opción a confusiones.
- Mantener la coherencia y formato en toda la app.
- Las imágenes ilustran la posición correcta, no la incorrecta.
- Dibujos y esquemas de trazos sencillos y con pocos detalles.
- Imágenes no saturadas de información.
- Único tipo de letra.

- Tamaño de letra suficientemente grande, entre 12 y 16 puntos, siendo una opción habitual 14 puntos.
- Tipografías claras (Arial, Calibri, Candara, Corbel, Gill Sans, Helvética, Myriad, Segoe, Tahoma, Tiresias ó Verdana).
- Negrita y subrayados para destacar palabras.
- Evitar efectos tipográficos, como adornos, colores y sombras.
- La letra blanca sobre fondo de color es más difícil de leer que la letra negra sobre fondo blanco.
- tamaño de la letra). No es conveniente un interlineado demasiado amplio, porque la separación excesiva de líneas puede confundir.
- Cada línea debe tener una sola oración.
- Las líneas no superarán los 60 caracteres, y contendrán un mínimo de 5 palabras y un máximo de 15 a 20, de modo que las frases no sean ni muy cortas ni muy largas.
- Alinear el texto a la izquierda, no justificarlo a la derecha.
- Es importante mantener un ritmo regular en la composición del texto en párrafos, ya que así aparecerá más nítido, ordenado y organizado visualmente.
- No partir una frase entre dos páginas.
- El número de líneas por página será limitado.
- Cuanto más blanco, mejor, pero siempre con texto en la página: utilizar amplios márgenes, blancos amplios en torno a los párrafos y líneas en blanco para distinguir párrafos y separar ideas.
- Evitar un diseño en columnas. Es mejor una composición horizontal que vertical.

Anexo 2)

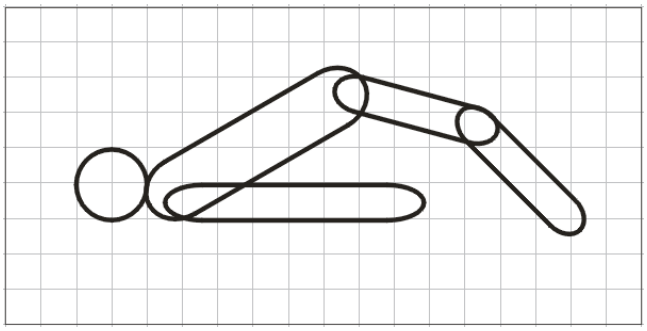
Algunas ideas para imágenes y dibujos de gimnasia

El aprendizaje de rutinas deportivas requiere al comienzo la presencia de un profesor que corrija errores en las posiciones e indique pautas generales. Posteriormente, el usuario, puede repetir los ejercicios de forma autónoma.

Las figuras de los ejercicios sirven para recordar algunos detalles generales. Incluso en la versión con accesibilidad, se ha decidido incluirlas con la idea que el usuario pueda, si lo considera oportuno, consultar a una persona con visión normal.

Se resumen algunas opciones de figuras: siluetas simplificadas de personas y fotografías sobre un maniquí de dibujo. Queda por estudiar utilizar o programar un maniquí articulado virtual de dibujo. La realización de fotografías de calidad profesional con un modelo humano real se ha descartado por las siguientes razones: requiere iluminación muy buena, alta disponibilidad de tiempo, varias personas.

Las siluetas simplificadas realizadas con un programa de dibujo permiten obtener rápidamente cualquier esquema de un ejercicio. La sensación de la imagen es fría y poco humana. La visibilidad es muy buena.



En las figuras siguientes se muestran algunos ejemplos de siluetas desarrolladas, que se descartaron de la aplicación final pero pueden ser útiles en algunas rutinas de gimnasia.

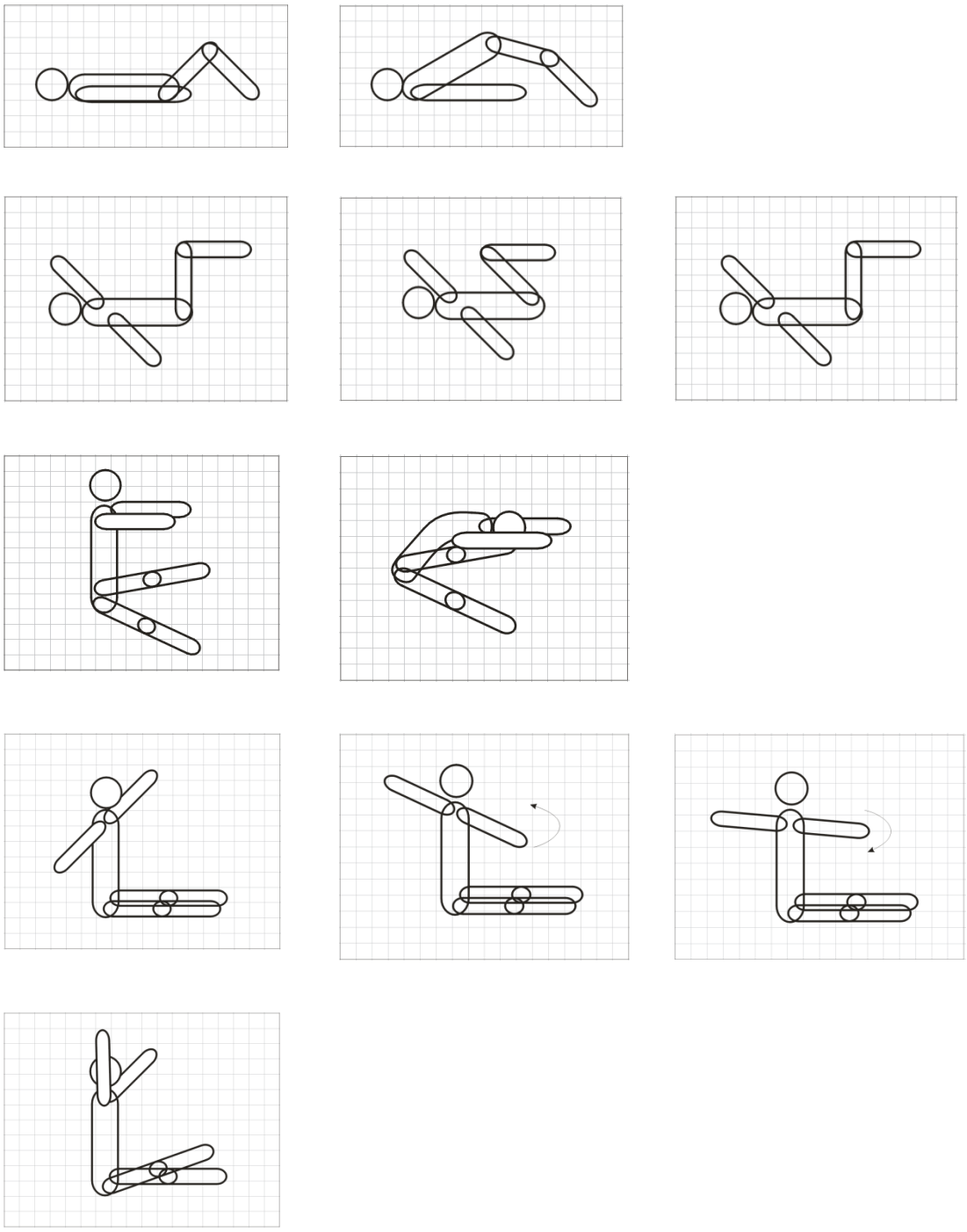


Figura 32 - Movimientos en siluetas

Las fotografías sobre un maniquí articulado de dibujo consiguen un aspecto más real y aportan cierta calidez y humor a las imágenes. La iluminación, escala y posición hacen que la obtención de estas fotos sea una tarea meticulosa y laboriosa, pero mucho menos que utilizando un modelo humano real.

Un inconveniente importante es que el maniquí no tiene todos los movimientos necesarios (por ejemplo, sentarse sobre los talones), por lo que es necesario en algunos casos utilizar dos maniquíes y realizar un retoque fotográfico posterior.

Las imágenes reales posteriormente se pueden transformar en dibujos con el objeto de aumentar la calidez o esconder defectos de iluminación.

Anexo 3)

PRESUPUESTO

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... 2.000 €
- Compra de un iPhone 6 ..... 600 €
- Material de oficina..... 150 €
- Total de ejecución material..... 2.750 €

2) Gastos generales

- 16 % sobre Ejecución Material..... 440 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material..... 165 €

4) Honorarios Proyecto

- 480 horas a 12 € / hora..... 5760 €

5) Material fungible

- Gastos de impresión..... 60 €
- Encuadernación..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto..... 9.375 €

7) I.V.A. aplicable

- 21% Subtotal Presupuesto ..... 1.969 €

8) Total presupuesto

- Total Presupuesto..... 11.344 €

Madrid, octubre de 2015

El Ingeniero Jefe de Proyecto

Fdo.: Óscar Javier Soria Andreu

Ingeniero de Telecomunicación

Anexo 4)

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de creación de una **Aplicación para movilidad articular y espalda sobre teléfono inteligente iPhone.** En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.