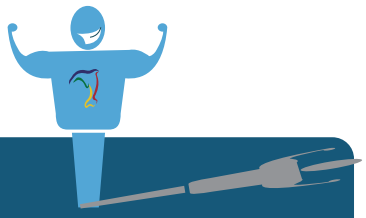
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

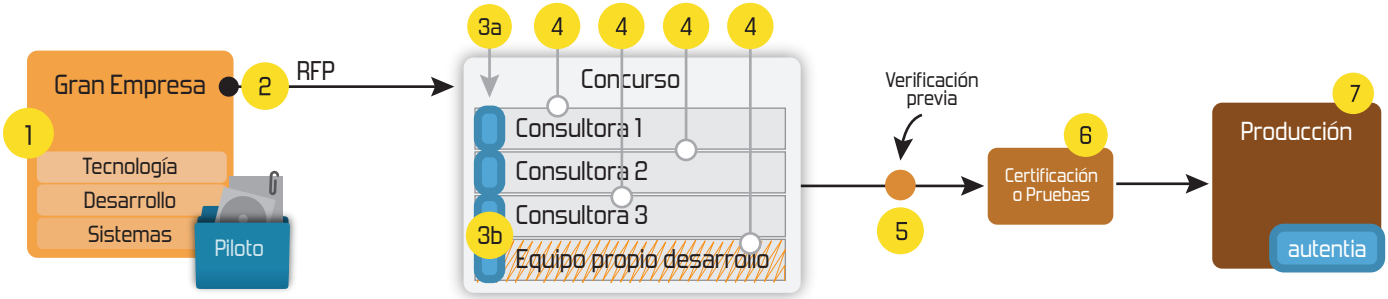
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

- Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery
- Gestor portales (Liferay)
- Gestor de contenidos (Alfresco)
- Aplicaciones híbridas
- Control de autenticación y acceso (Spring Security)
- UDDI
- Web Services
- Rest Services
- Social SSO
- SSO (Cas)
- JPA-Hibernate, MyBatis
- Motor de búsqueda empresarial (Solr)
- ETL (Talend)
- Dirección de Proyectos Informáticos.
- Metodologías ágiles
- Patrones de diseño
- TDD
- Tareas programadas (Quartz)
- Gestor documental (Alfresco)
- Inversión de control (Spring)
- BPM (jBPM o Bonita)
- Generación de informes (JasperReport)
- ESB (Open ESB)

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

Ignacio Acisclo Pérez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE

Ver todos los tutoriales del autor

Fecha de publicación del tutorial: 2014-11-20

Tutorial Apple Watch

0. Índice de contenidos.

1. Entorno

Este tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Mac Book Pro 15" (2,5 Ghz Intel Core i7, 16 GB DDR3)
- Sistema Operativo: Mac OS X Yosemite
- Xcode 6.2 beta

2. Introducción

Por fin Apple ha facilitado a los desarrolladores el SDK para el esperado reloj de la compañía. Ayer liberaron la beta de Xcode 6.2 con la inclusión de esta posibilidad, aunque hay que admitir que no lo han facilitado completo: A día de hoy con este kit de desarrollo solo podemos construir extensiones de nuestras app de iPhone para el reloj y no una app independiente que no requiera la ejecución de una app “padre” en nuestro teléfono.
Es decir, en realidad es prácticamente lo mismo que cuando creamos un widget en el centro de notificaciones, que al final la lógica de la app, sus controladores, modelos... etc “corre” en la app principal y la interfaz de usuario en nuestro reloj / centro de notificaciones.

Vamos a disponer de 3 tipos de aplicaciones/estilos para el reloj;

- **Glances:** Que básicamente es una extensión de nuestra app sin interactividad, solo para mostrar al usuario información y en caso de pulsar sobre cualquier sitio nos abre la app padre en el teléfono. El concepto de este estilo es el mismo que los widgets.
- **Actionable Notifications:** Se trata de un tipo de extensión pensada para informar al usuario y esperar una respuesta por su parte. Consta de dos estados, vistazo rápido y vistazo lento. En el lento se muestra la minima información posible, solo un indicador del evento que ha sucedido. En caso de no mantener el usuario la muñeca en alto, se quedaría en reposo. En caso de mantener la muñeca en alto cambia a estado vistazo rápido y entonces podremos mostrar la información detallada así como botones para interactuar con la notificación.
- **WatchKit App:** Es más genérica, podemos hacer todo lo que este dentro del marco de watchKit.

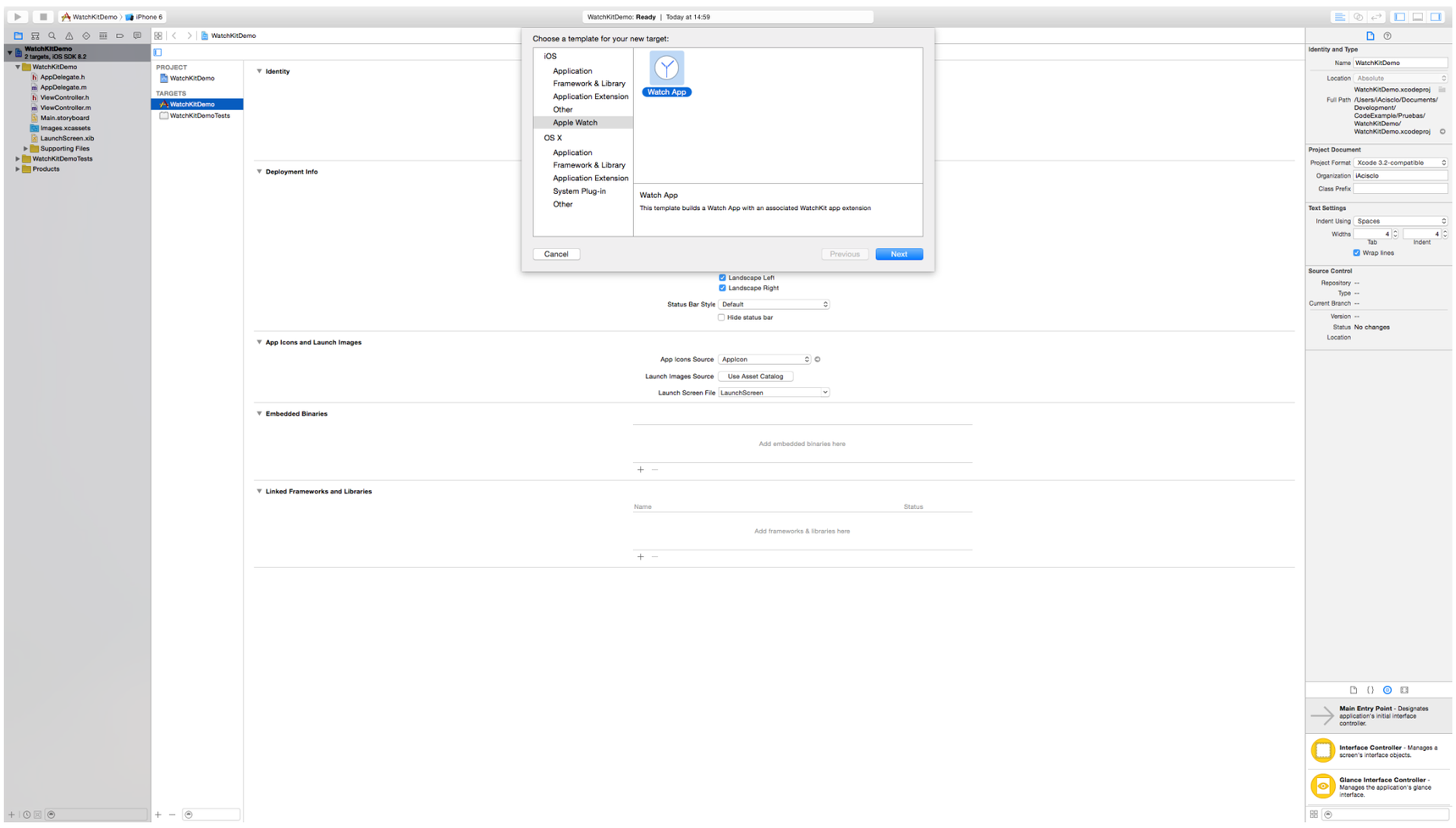
De cara a la navegación de nuestra watchApp vamos a disponer de dos tipos de controladores, el típico NavigationController que se basa en la navegación jerárquica y un Page-Based que mostrará un controlador por “página”. Es importante destacar que no se pueden combinar, pero es posible presentar modalmente un controlador.

El autolayout juega un papel importante debido al reducido tamaño del que disponemos, pero no es como al que estamos acostumbrados. Ahora por ejemplo en un contenedor que ocupe el ancho de la pantalla nos permiten elegir entre 3 tipos de alineaciones para sus elementos: izquierda, derecha y centro. De hecho Apple nos indica que limitemos a 3 el número de

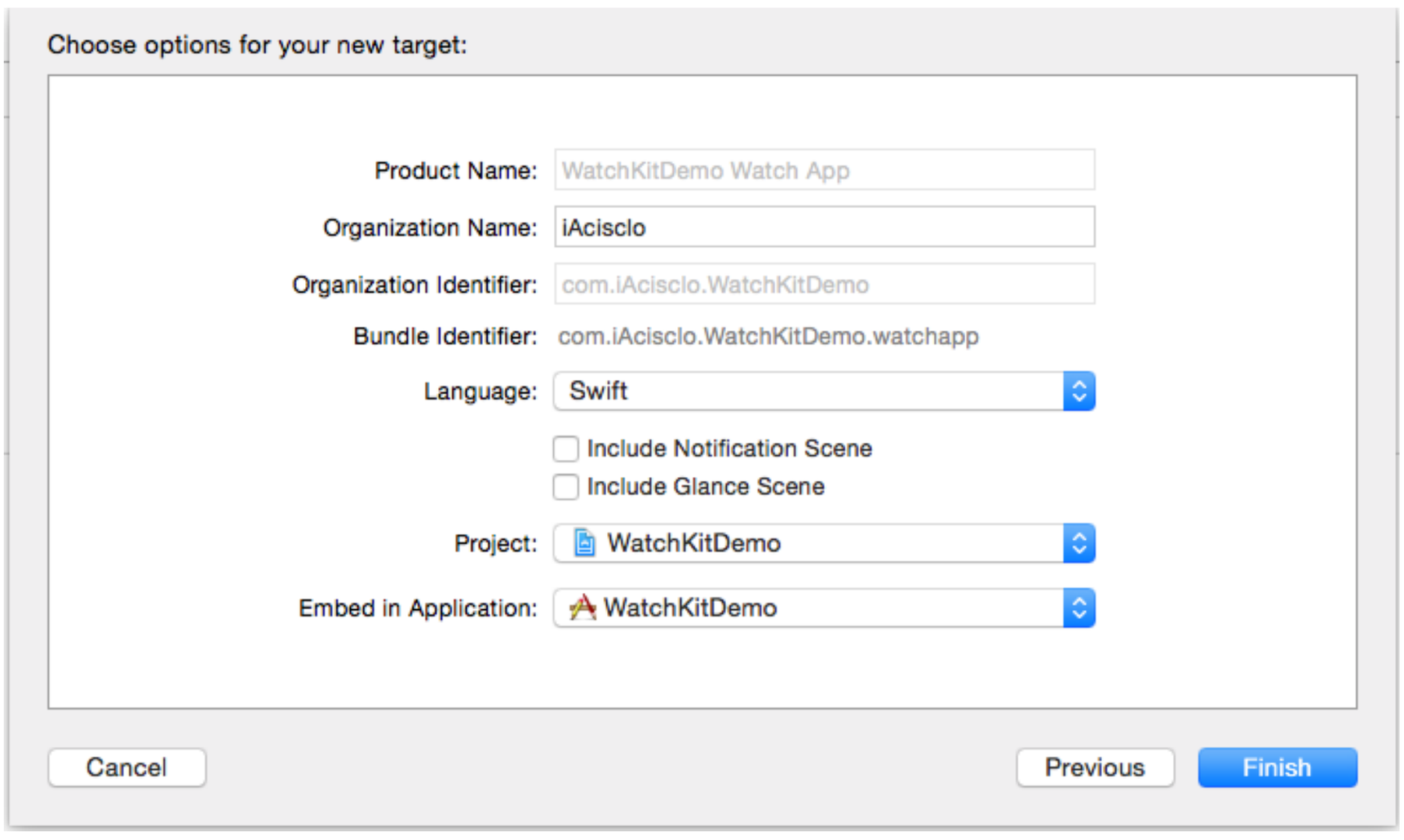
elementos que podemos colocar horizontalmente.
Tampoco hay gestureRecognizers, el sistema detecta nuestros gestos y hace lo predeterminado, por ejemplo en un page-based pide el controlador siguiente o anterior.

3. Desarrollo de WatchApp

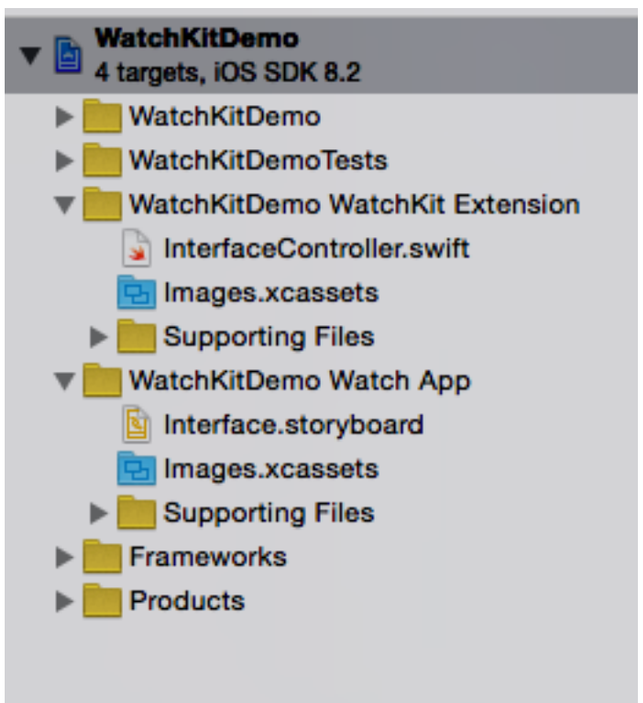
Visto un poco de teoría, vamos a hacer una WatchKit App, así que una vez descarguemos la beta de Xcode 6.2 podremos empezar.
Vamos a iniciar un nuevo proyecto al que llamaremos WatchKitDemo y seleccionamos la plantilla “Single View Application”.
Lo dejamos tal y como nos viene por defecto y seleccionamos nuestro proyecto para poder añadir un nuevo target en el botón “+” de la parte inferior izquierda:



Seleccionamos Apple watch y pulsamos siguiente, después vamos a deseleccionar las dos opciones que vienen de Glance y Notificaciones ya que hoy vamos a ver como manejar la extensión WatchApp y por supuesto seleccionamos Swift ;-)

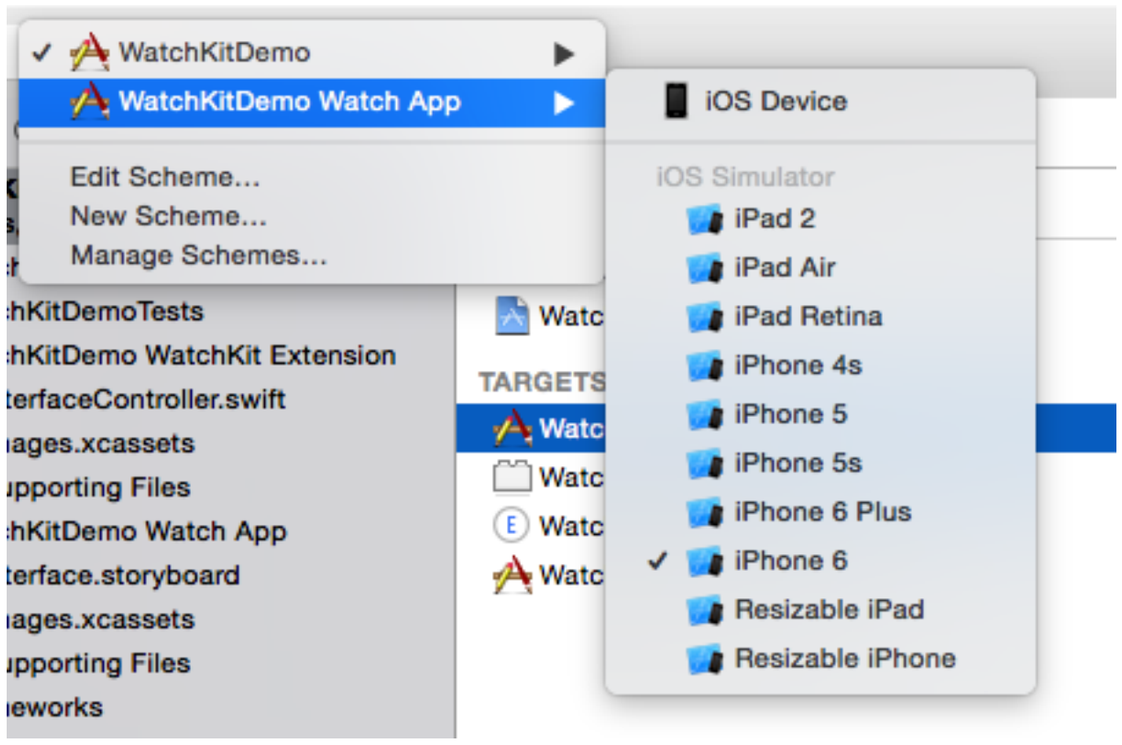


Como podeis ver, se han generado en el proyecto dos directorios nuevos.

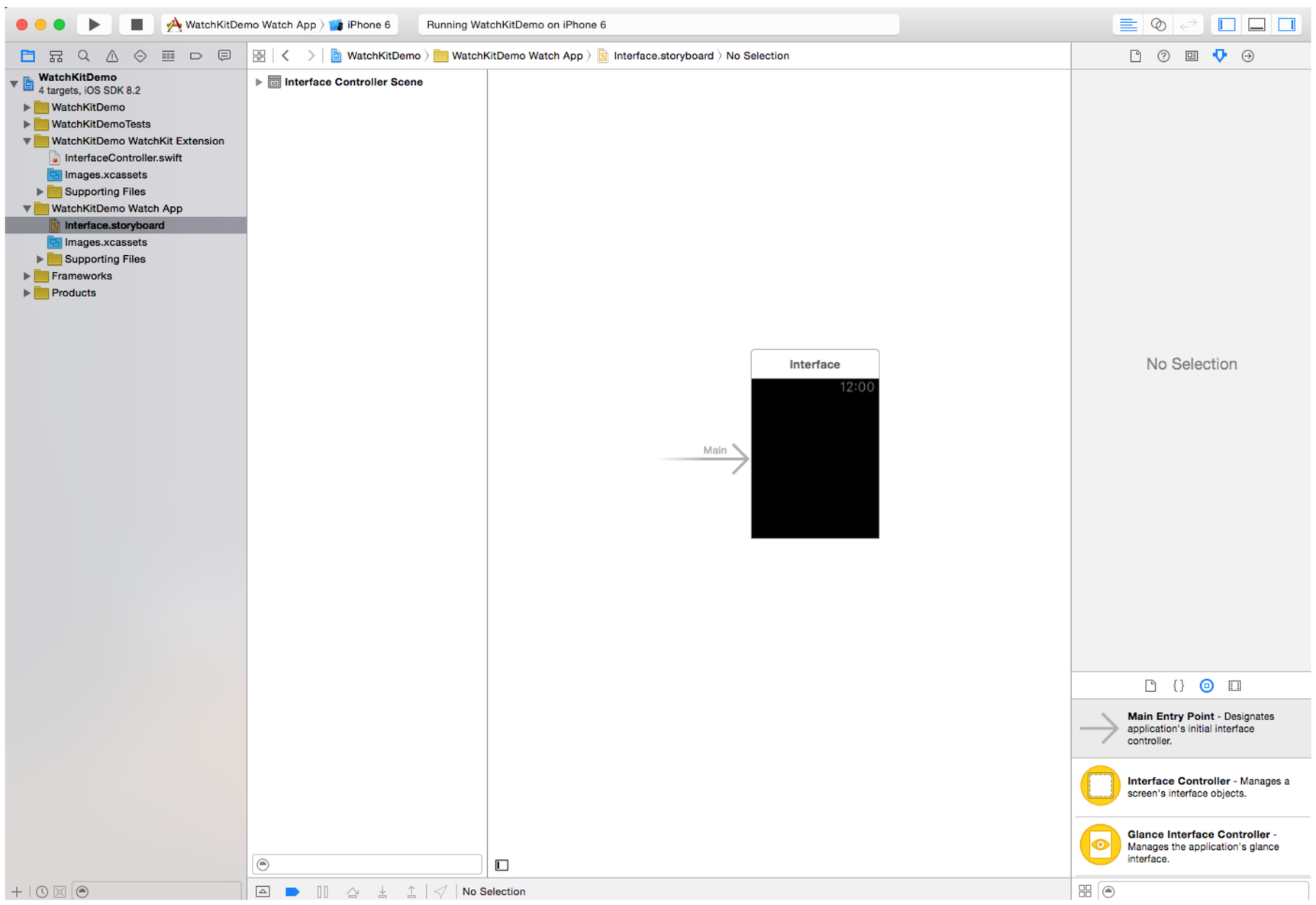


La carpeta WatchKit Extensión corresponde a la parte modelo y controlador, que es la parte que irá en nuestro teléfono y la carpeta Watch App es la que va en nuestro reloj.

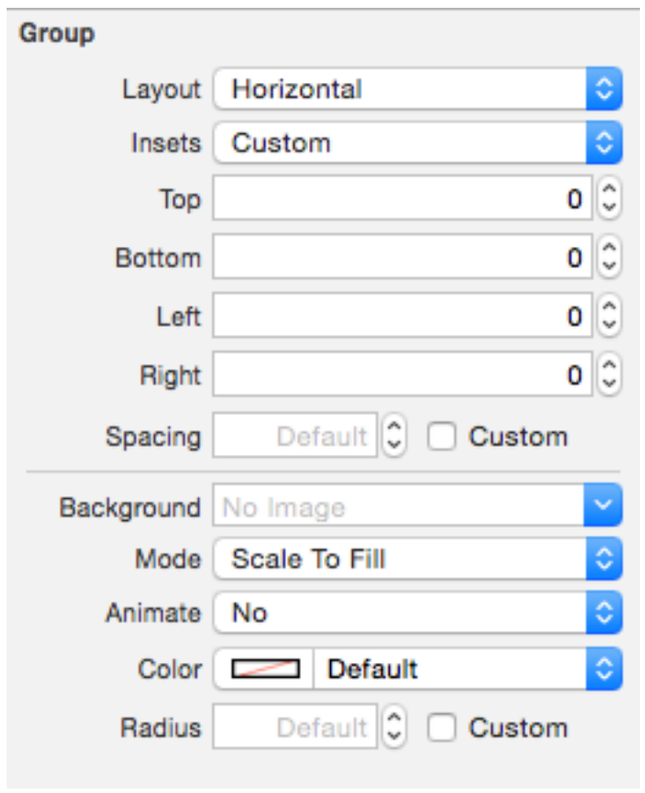
Xcode nos ha creado ya nuestro esquema para poder ejecutar nuestra WatchApp:



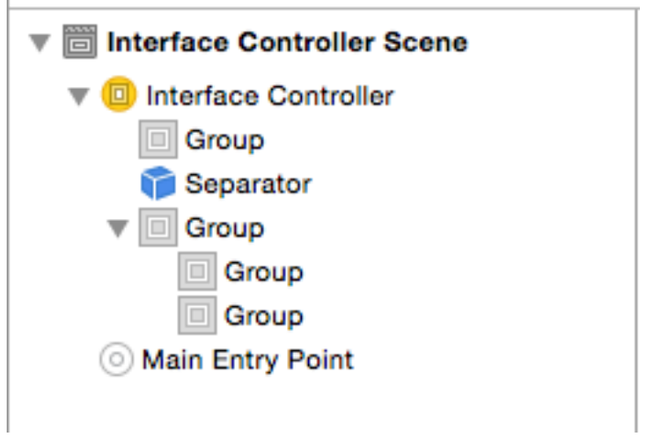
Ahora vamos a nuestro storyboard que ha creado Xcode en la carpeta Watch app:



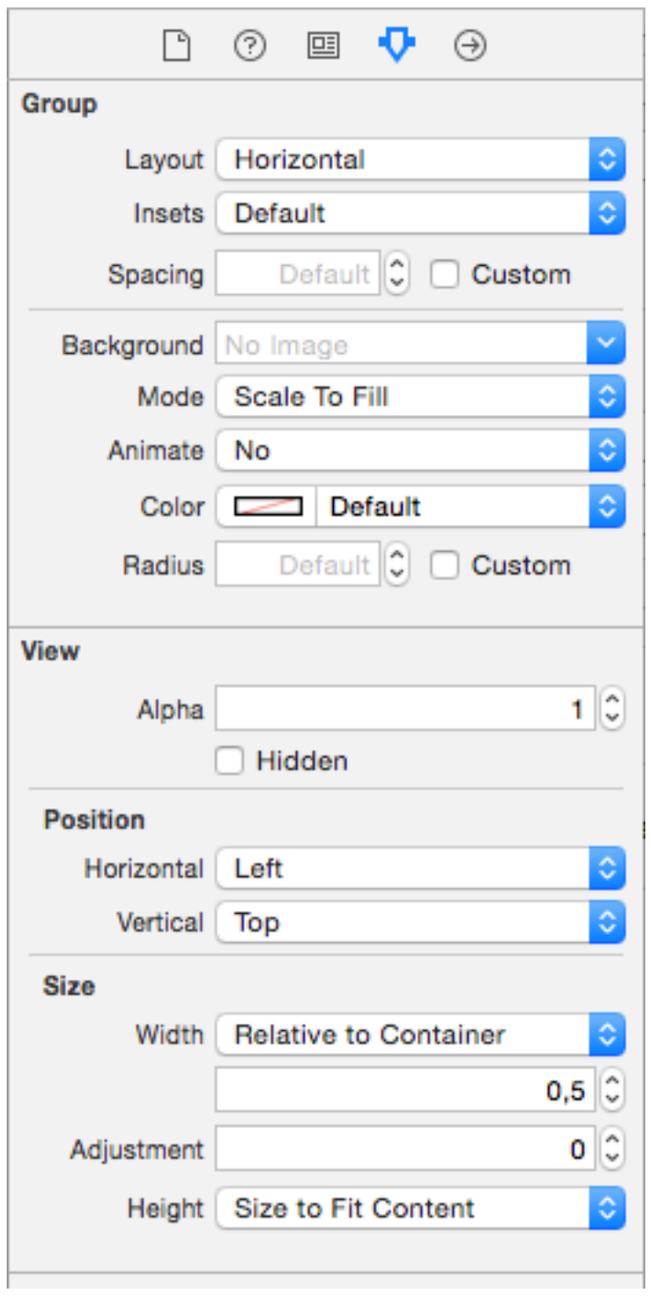
En la parte inferior derecha tenemos nuestro catálogo de objetos que podemos añadir a nuestras vistas. Es importante comentar el elemento “group”, que es básicamente un contenedor. Podemos colocar varios contenedores en nuestras vistas e incluso contenedores que contienen otros para conseguir el layout adecuado. Los contenedores tienen opción de distribuir los elementos de su interior de forma horizontal o vertical, tambien podemos darles un margen custom para cada lado así como un corner-radius y otras opciones interesantes, aunque limitadas.



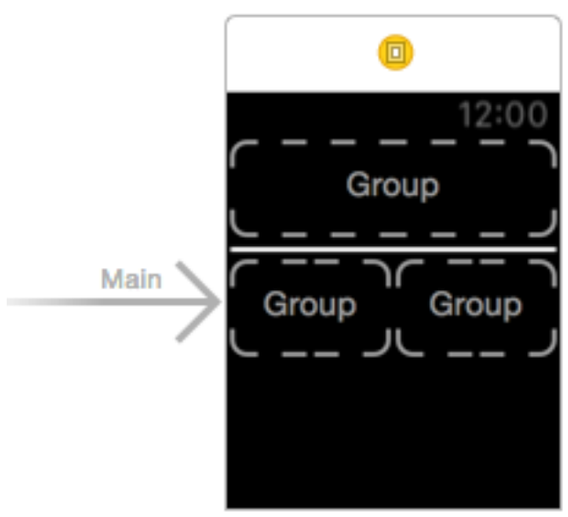
Tambien tenemos un elemento Separador para poder dividir nuestro layout en diferentes secciones. Vamos a introducir en nuestro controller dos groups y un separator a nivel superior y dentro del segundo group arrastramos otros dos groups, quedando el árbol de elementos de esta forma:



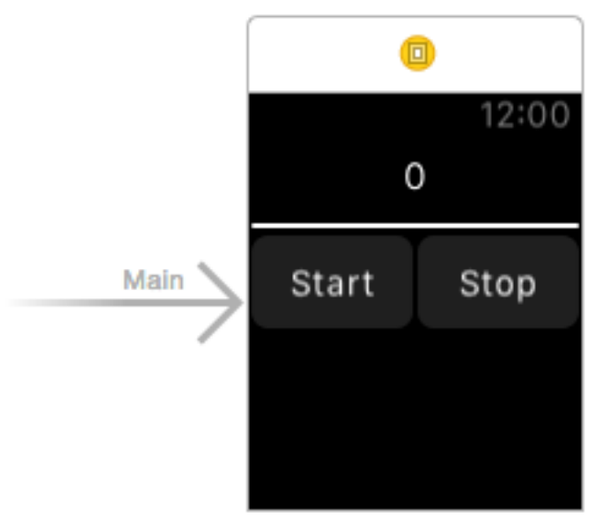
Para que los dos groups que están dentro del segundo nos queden correctamente alineados horizontalmente tenemos que seleccionarlo y dejar su inspector de atributos con la siguiente configuración:



La posición debemos dejarla en concordancia con la ubicación del group, de tal forma que nos quede una cosa así:



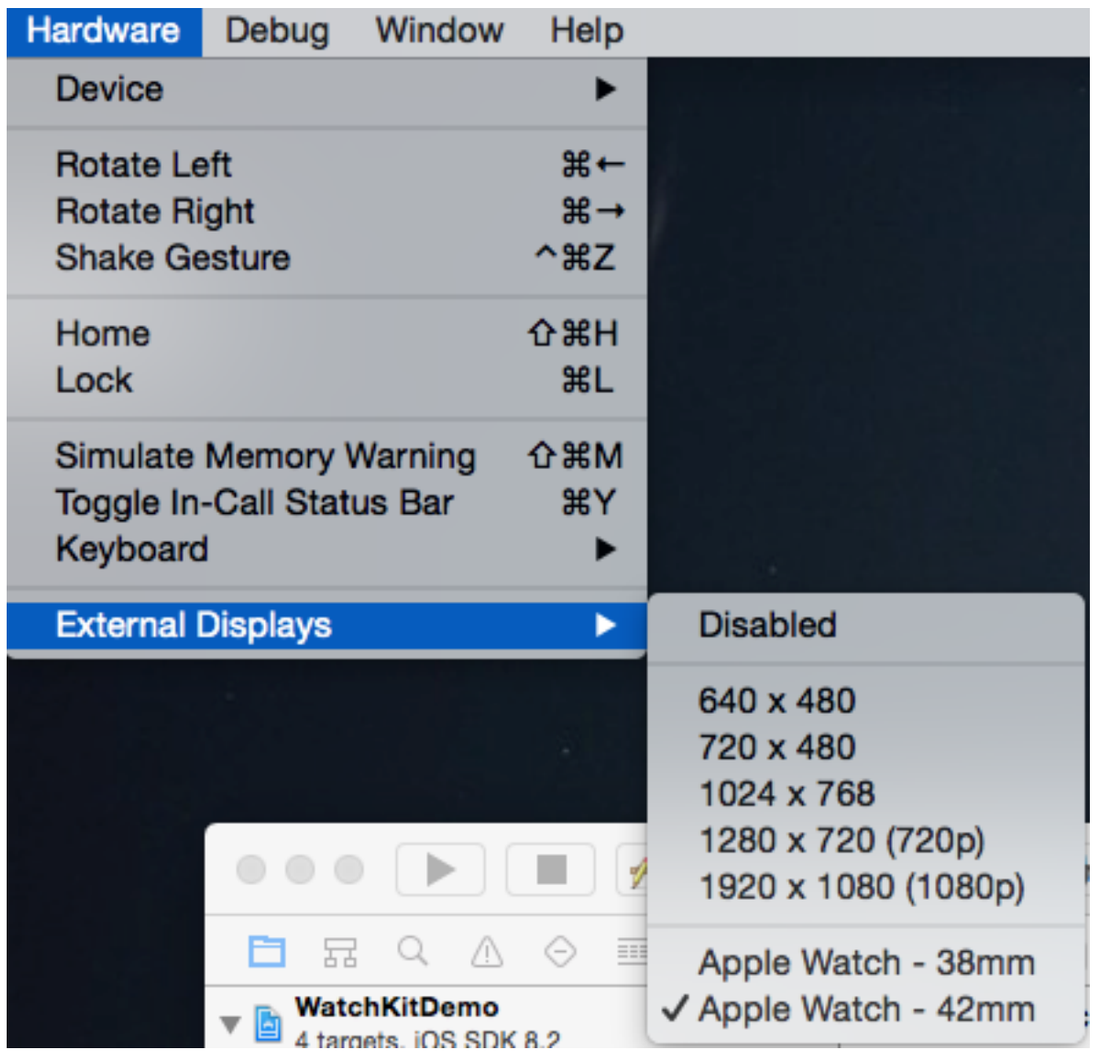
Ahora vamos introducir un objeto Timer en el primer group y un botón en cada uno de los groups pequeños:



Al timer vamos a darle un alto y ancho fijo para que el container se ajuste correctamente, y a los botones les ponemos los textos “start” y “stop”. A continuación generamos sus variables en nuestro controlador, que nos quedaría de la siguiente forma:

```
import WatchKit

class InterfaceController: WKInterfaceController {

    @IBOutlet weak var timer: WKInterfaceTimer!

    @IBAction func start() {

        timer.start()
    }

    @IBAction func stop() {

        timer.stop()
    }

    override init(context: AnyObject?) {
        super.init(context: context)
    }

    override func willActivate() {
        super.willActivate()

        timer.setDate(NSDate(timeIntervalSinceNow: 180))
        timer.start()
    }

    override func didDeactivate() {
        super.didDeactivate()
    }

}
```

Y con este cambio ya podemos pulsar command+R para compilar y ejecutar. En caso de no nos aparezca la pantalla del reloj nos vamos al menu del simulador y seleccionamos la opción external displays y ya está, tenemos nuestra primera WatchApp lista.



Podéis descargaros el proyecto de ejemplo del siguiente enlace.