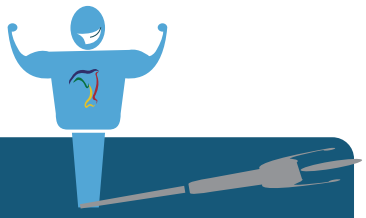
autentia

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...

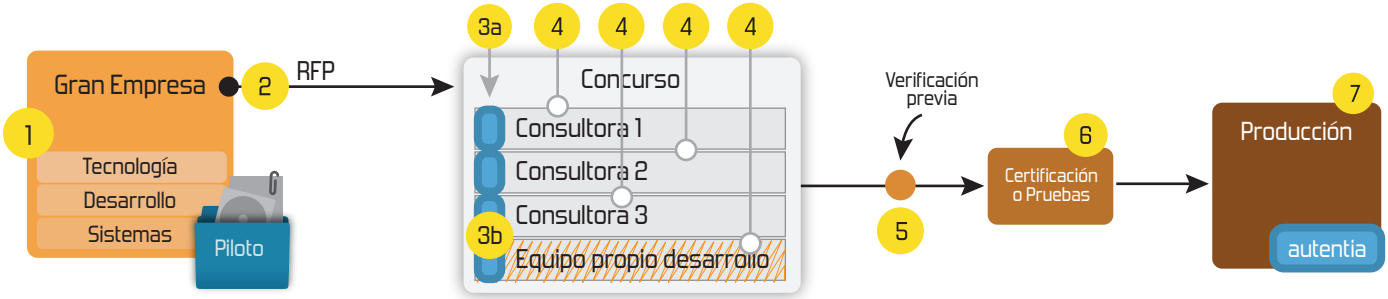
1. Desarrollo de componentes y proyectos a medida



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: 2015-02-13

Tutorial visitado 929 veces Descargar en PDF

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

Se acerca el momento de comprar nuestros relojes de la manzana, y no se vosotros pero yo estoy deseando poder probar mis aplicaciones en él.

Con motivo de celebración del primer curso online de Autentia vamos a ver un tutoría de como construir una tabla.

Las tablas en WatchKit están representadas por el objeto WKInterfaceTable , en nuestro Storyboard, cuando arrastramos una tabla al controlador , en la jerarquía de vistas vamos a ver nuestra tabla y dentro un objeto subclase de NSObject que Xcode ha nombrado automáticamente como Table Row Controller, vamos a necesitar generar nuestra subclase de NSObject para manejarlo y un identificador que pondremos en el inspector de atributos del storyboard.

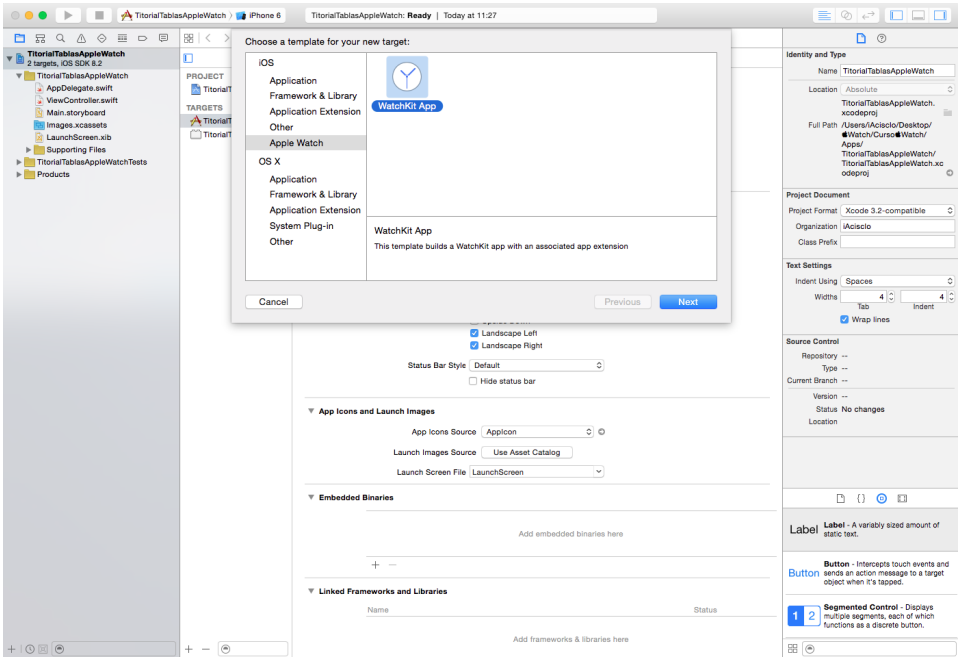
Esta va ser nuestra plantilla para las celdas que generemos bajo el identificador designado, dentro del table row controller, tendremos un objeto WKInterfaceGroup para poder hacer el diseño de la celda a nuestro antojo.

A la hora de indicar cuantas celdas va a disponer nuestra tabla inicialmente tenemos que tener en cuenta Si únicamente vamos a utilizar un tipo de celda, en cuyo caso usaremos la función setNumberOfRows(withRowType:) o, en caso de usar varios tipos de celda usaremos la función setRowTypes() en el que le pasamos un array con los tipos. La longitud de este array equivale al numero de filas que tendrá la tabla coincidiendo cada tipo, con la posición.

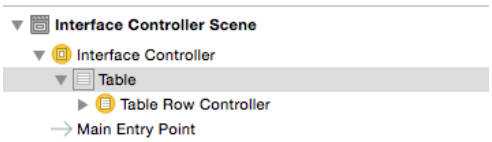
Las tablas pueden cambiar su contenido de forma dinámica, es decir, pueden cambiar la información que contienen sus celdas o variar el numero de ellas en tiempo de ejecución. Las funciones que usaremos para dicha tarea son insertRowsAtIndexes(withRowType) y removeRowsAtIndexes().

3. Desarrollo de la tabla.

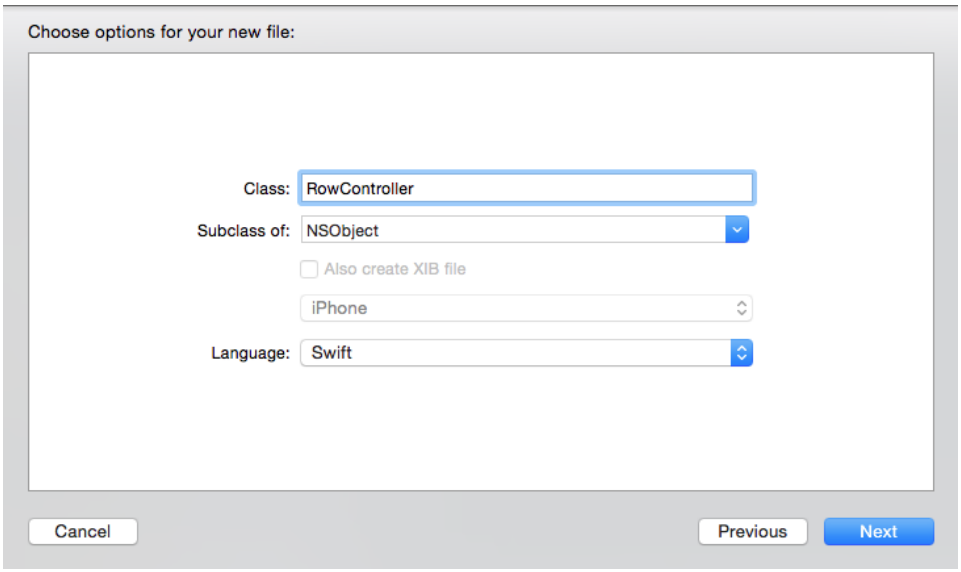
Si habéis visto el último tutorial sobre watchKit, ya sabéis como iniciar un nuevo proyecto y añadir el target para nuestra aplicación en WatchKit. Así que creamos un nuevo proyecto en Xcode y añadimos el target.



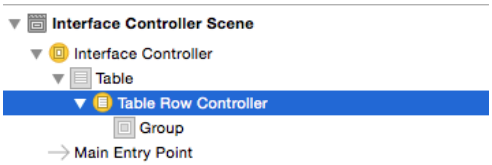
Vamos a arrastrar una tabla a nuestro interface.storyboard para que nuestro controlador de entrada quede de la siguiente forma:



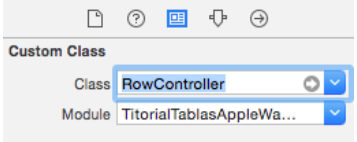
Ahora vamos a crear una subclase NSObject que llamaremos RowController.



Por ultimo vamos a nuestro storyboard y seleccionamos Table Row Controller,

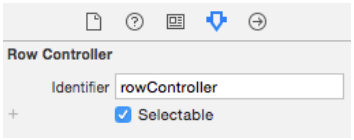


una vez seleccionado nos vamos al inspector de identidad y le decimos la clase a la que pertenece.

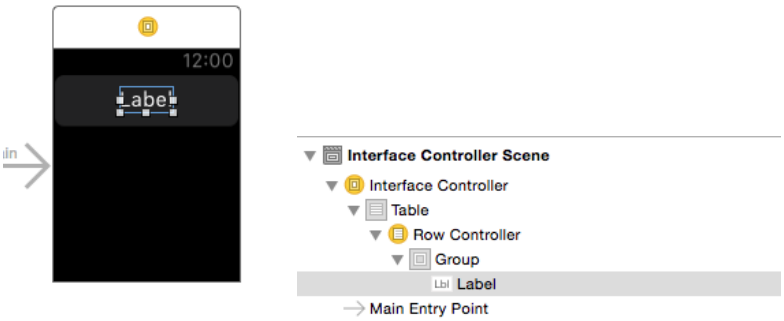


También le damos un identificador en el inspector de atributos:

Últimos Tutoriales del Autor



Vamos a meter un label en nuestra “plantilla” de celda , así que arrastramos un objeto WKInterfaceLabel al group que se encuentra en el RowController;



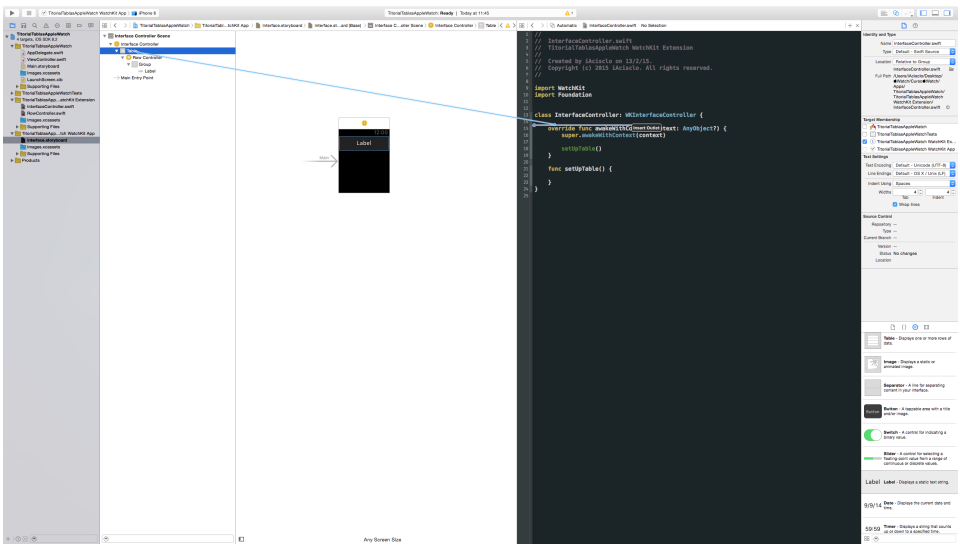
y conectamos mediante un Outlet a nuestra clase.

```
import WatchKit

class RowController: NSObject {

    @IBOutlet weak var textLabel: WKInterfaceLabel!
}
```

Ahora vamos a ir a nuestra clase InterfaceController y creamos la función setUpTable() y la llamamos desde nuestra función awakeWithContext()

```
import WatchKit
import Foundation


class InterfaceController: WKInterfaceController {

    override func awakeWithContext(context: AnyObject?) {
        super.awakeWithContext(context)

        setUpTable()
    }

    func setUpTable() {

    }
}
```

Vamos a necesitar un outlet de nuestra tabla así que hacemos la misma operación que con el Label en nuestra clase RowController, creamos el outlet de y lo conectamos en el storyboard



Una vez conectado vamos a nuestro controlador y creamos un Array que nos va a servir de modelo;

```
let model = ["celda 1","celda 2","celda 3"]
```

Y por ultimo en nuestra función setUpTable() vamos a incluir el siguiente código;

```
func setUpTable() {

        table.setNumberOfRows(countElements(model), withRowType: "rowController")
        for (var i: Int = 0; i < countElements(model); i++) {
            let row = table.rowControllerAtIndex(i) as RowController
            row.textLabel.setText(model[i] as String)
        }
    }
```

Si compilamos y ejecutamos veremos que nuestra pantalla del simulador de Apple Watch mostrará la tabla con 3 celdas.

4. Conclusión

Como veis la creación de una tabla es algo bastante sencillo con WatchKit, pero si queréis aprender más sobre ellas y el resto de componentes que forman esta librería podéis acceder al curso online que hemos preparado en Udemy.