GRADO EN INGENIERÍA INFORMÁTICA DE GESTIÓN Y SISTEMAS DE INFORMACIÓN

TRABAJO FIN DE GRADO

PLATAFORMA IOT PARA EL REGISTRO DE DATOS METEOROLÓGICOS Y SU VISUALIZACIÓN MEDIANTE UNA APLICACIÓN MÓVIL

DOCUMENTO III: MANUAL IONIC

Alumno: Saavedra González, Alexander

Director (1): Casquero Oyarzabal, Oskar

Director (2): Equiraun Martínez, Harkaitz

Curso: 2017-2018

Fecha: Bilbao, a 23 de febrero de 2018

ANEXO II: Manual Ionic

Descripción general de Ionic

Ionic es un framework para el desarrollo de aplicaciones híbridas, inicialmente pensado para móviles y tablets, aunque ahora también capaz de implementar aplicaciones web e incluso dentro de poco, aplicaciones de escritorio multiplataforma. Su característica fundamental es que usa por debajo Angular y una cantidad de componentes enorme, que facilita mucho el desarrollo de las aplicaciones.

Se trata de una estupenda herramienta para la creación de aplicaciones sorprendentes, pensada para obtener resultados de una manera rápida y con una menor inversión económica, ya que permite crear aplicaciones para distintas plataformas móviles con una misma base de código.

Qué es una aplicación híbrida

Una aplicación híbrida es aquella que permite desarrollar aplicaciones para móviles en base a las tecnologías web: HTML + CSS + JavaScript. Son como cualquier otra aplicación de las que puedes instalar a través de las tiendas de aplicaciones para cada sistema, por lo que en principio usuarios finales no percibirán la diferencia con respecto a otros tipos de aproximaciones diferentes, como las aplicaciones nativas.

Al ejecutarse con tecnologías web, los desarrolladores que ya tienen experiencia en el desarrollo en este medio pueden aprovechar sus conocimientos para lanzarse de una manera más rápida en el desarrollo de aplicaciones para móviles. Para hacer esto posible, las aplicaciones híbridas se ejecutan en lo que se denomina un *web view*, que no es más que una especie de navegador integrado en el móvil y en el que solamente se ejecuta la aplicación híbrida.

Las aplicaciones híbridas son interesantes por diversos motivos:

- Con una misma base de código serán capaces de compilar aplicaciones para funcionar correctamente en una gran cantidad de sistemas operativos de móviles o tablets. Generalmente nos será suficiente que nuestra aplicación funcione en iOS y Android, pero Ionic es capaz de compilar a otros sistemas como Windows Phone.
- El coste del desarrollo es sensiblemente menor, ya que no es necesario contar con varios equipos de desarrollo para cada lenguaje concreto de cada plataforma.
- El tiempo de desarrollo también es menor, ya que solo es necesario construir la aplicación una vez e inmediatamente la tendremos en todas las plataformas a las que nos dirigimos.
- Es de más fácil adaptación para los desarrolladores que vienen de la web.

Pero todo no son ventajas, también podemos enumerar algunas desventajas en relación con las aplicaciones nativas (aquellas construidas con los lenguajes propios de cada sistema, Java para Android y Swift / Objective-C para iOS):

- El rendimiento de una aplicación nativa suele ser mejor que el de una híbrida, aunque las híbridas han mejorado mucho en este sentido.
- Al ejecutarse en un *web view* dependemos de las tecnologías disponibles para el desarrollo web, que pueden ser menos ricas y potentes que las disponibles en nativo.
- En las aplicaciones nativas trabajamos directamente con el hardware del teléfono, mientras que en las híbridas dependemos de *plugins* para su acceso. Esto puede derivar en problemas, pero afortunadamente cada vez son menores.

Integración con Angular

Ionic está desarrollado sobre el framework JavaScript Angular. Esto quiere decir que para el desarrollo con Ionic podemos apoyarnos en todas las ventajas de desarrollo con Angular, lo que nos permitirá contar con una excelente estructura de proyecto, uso de patrones de diseño de software y una buena gama de componentes y directivas.

Nuestro código por tanto tendrá más calidad usando Angular y podremos usar muchas de sus utilidades pensadas para un desarrollo más rápido. Gracias a usar Angular, tenemos además la certeza de disponer de un código más optimizado y de mayor rendimiento, adaptado para el momento actual y el futuro.

Desarrollo basado en componentes

En Ionic, trabajamos en base a componentes. Esto quiere decir que nuestras aplicaciones serán compuestas por un árbol de componentes que se utilizan los unos a los otros para la conclusión de los objetivos globales de la aplicación.

Los componentes están pensados para, de manera modular y encapsulada, resolver pequeños problemas. Por ejemplo, puede haber componentes para implementar un sencillo botón, componentes para hacer un sistema de navegación por *tabs*, para selectores de fechas, etc. Integrando componentes somos capaces de construir aplicaciones grandes y complejas. Ionic nos ofrece ya de base una cantidad muy grandes de componentes que son capaces de trabajar perfectamente en dispositivos móviles con pantallas táctiles, pero obviamente para el desarrollo de nuestras aplicaciones necesitaremos construir nuestros propios componentes que implementen los comportamientos más específicos de nuestro modelo de negocio.

TypeScript

Otra cosa que viene dada por el desarrollo de Angular es el uso del lenguaje TypeScript, que no es más que un "superset" de JavaScript. Dicho de otra forma, TypeScript es JavaScript, pero con añadidos pensados para mejorar el trabajo por parte de los desarrolladores, haciéndonos más productivos.

La mayor aportación de TypeScript al lenguaje JavaScript es la posibilidad de definición de tipos para las variables, pero en general aporta mucho más y además nos permite usar todas las mejoras de ES6 y algunas de ES7 en las aplicaciones.

Los componentes de Ionic ya vienen adaptados al dispositivo de manera estética. Quiere decir que, cuando se compila una aplicación para iOS el componente se visualizará de manera diferente que cuando se compila para Android. En Android usará Material Design mientras que en iOS usará las guías de diseño definidas por Apple.

Esto es una ventaja en sí, porque las personas disfrutarán de aplicaciones con una experiencia de usuario cercana a la que están acostumbrados en su teléfono y nos evita a los desarrolladores la necesidad de trabajar más para conseguir este efecto. Sin embargo, como autores de las aplicaciones con Ionic también somos capaces de alterar el diseño de las aplicaciones, proporcionando una experiencia de usuario específica y original para nuestra propia aplicación.

Desarrollo y compilado de aplicaciones

Con Ionic desarrollamos aplicaciones con las tecnologías web. Durante toda la etapa de desarrollo usaremos el navegador para visualizar las aplicaciones, lo que permite un flujo de trabajo muy productivo, ya que no se tiene que compilar. Aunque lo cierto es que, por la necesidad de transpilar el código de TypeScript a JavaScript compatible con el navegador, no será tan rápido de visualizar los cambios como simplemente refrescar la página del navegador.

Una vez que la aplicación está construida se tiene que realizar el proceso de compilación, en el que se producen los ejecutables específicos para cada dispositivo. Ese proceso sí es un poco más pesado para el procesador, pero se hace solo cuando tenemos que lanzar una nueva versión que se subirá a las tiendas de aplicaciones.

Apache Cordova

Ionic se basa también en Apache Cordova para la implementación de las aplicaciones. Hay partes, como el acceso a los componentes nativos del dispositivo, en las que se usan *plugins* que nos proporciona Apache Cordova principalmente. Actualmente también Ionic es proveedor de algunos *plugins* para trabajo con la parte nativa del teléfono. Aquí como nativo se refiere a elementos como la cámara, acelerómetro, teclado virtual, etc. Todos esos elementos se pueden usar desde las aplicaciones de Ionic, con los correspondientes *plugins* nativos, que forman una especie de puente entre el desarrollo con JavaScript y el teléfono.

Apache Cordova también es el software que nos permite compilar el desarrollo realizado con Ionic con tecnologías web en aplicaciones para móviles instalables vía tiendas de aplicaciones.

El intérprete de línea de comandos de Ionic

Ionic CLI es el intérprete por línea de comandos de Ionic, que contiene una serie de herramientas útiles para realizar, de una forma sencilla, muchas tareas habituales en el desarrollo y producción de aplicaciones con Ionic.

Iniciar aplicaciones Ionic

ionic start nombre-proyecto

Lo que conseguimos con este comando es crear una carpeta llamada "nombre-proyecto" que iniciará una aplicación Ionic. Esto incluye una cantidad de subcarpetas con una estructura de proyecto definida para las aplicaciones de Ionic. Más adelante iremos conociendo esa estructura con detalle.

Templates disponibles en Ionic

Lo interesante es la lista de *templates* que tenemos disponibles para crear aplicaciones, con distinto código base. Nuestras aplicaciones, lejos de comenzar con una página en blanco, pueden crearse en base a distintos tipos de *templates*, con estructuras de páginas diferentes.

Si ejecutamos el siguiente comando tendremos una lista de los *templates* disponibles en nuestro sistema:

ionic start --list

Encontrarás varias opciones para comenzar tus aplicaciones, desde una aplicación en blanco, una con *tabs*, con un completo menú lateral, etc. Sin embargo, estos no son todos los que podrías usar, también existen diversos starters que se encuentran disponibles en Ionic Market, tanto gratuitos como de pago.

Además, puedes encontrar repositorios en GitHub donde también se encuentran aplicaciones creadas con Ionic, que te pueden dar un buen punto de partida para aprender examinando el código del proyecto.

Correr un proyecto de Ionic en el navegador

Por ser aplicaciones híbridas, realizadas con JavaScript, Ionic tiene una ventaja fundamental a la hora de desarrollar apps: permitir desarrollar usando el navegador como entorno de ejecución. Esto es muy interesante, ya que nos permite un flujo de desarrollo muy rápido entre la escritura de código y la visualización de los cambios. Además, nos evita tener que instalar muchos programas para comenzar, emuladores, entornos de desarrollo, programas para llevar a producción, etc.

Para mostrar nuestra aplicación en el navegador simplemente tenemos que iniciar un servidor web integrado en Ionic CLI, con el comando "serve".

Este comando lanza todo el proceso de transpilación y abre la aplicación en el navegador. Además, contiene un sistema de "LiveReload", que permite refrescar el navegador cuando haya cambios en el código de la aplicación.

Hay opciones interesantes que podemos usar para personalizar el servidor:

- ***--lab:*** Nos permite abrir el "ionic lab" con el que podemos ver cómo se presentaría la aplicación en cada tipo de plataforma móvil.
- ***--port:*** si queremos cambiar el puerto donde escucha el servidor.
- ***--address:*** Si indicas esta opción, asignando tu IP local, el servidor se iniciará sobre esa IP en lugar de localhost. Esto puede ser útil si quieres visualizar el contenido desde otros sistemas, en caso de que no lo consigas ver desde ordenadores o dispositivos externos con "ionic serve" a secas.
- ***--platform:*** Esto te permite iniciar viendo la aplicación tal como se mostraría en una plataforma en concreto (iOS / Android).

Otros comandos de Ionic CLI

- ***ionic run:*** Permite ejecutar la aplicación Ionic que estamos desarrollando en un dispositivo conectado al ordenador con el que estamos trabajando.
- ***ionic platform:*** Sirve para añadir una nueva plataforma para realizar el *build* de una aplicación.
- ***ionic build:*** Genera la aplicación para una plataforma especificada (ej: ionic build android), esta parte se realiza gracias a Cordova.
- ***ionic emulate:*** lanza el emulador para visualizar en él nuestra aplicación Ionic. Es simular a un ionic serve, solo que usa el emulador disponible en lugar del navegador.
- ***ionic generate:*** este es un comando muy extenso, que a su vez tiene una serie de sub comandos para generar distintos tipos de componentes en un proyecto Ionic, como pueden ser páginas, *providers*, *pipes*, etc.

Al crear un proyecto con Ionic CLI obtenemos una lista de carpetas y archivos que forman parte de nuestra aplicación, y cuya parte tiene su función.

Carpeta src

En esta carpeta es donde realizamos la mayor parte del trabajo con Ionic, son archivos fuente con el código original que escribirás al desarrollar tu aplicación, o con código que viene dado al iniciar el proyecto con "ionic start". "src" contiene código que se tendrá que transformar para poder ejecutarse o producir los archivos finales que se irán a las *stores*.

El 98% de tu proyecto lo vas a pasar dentro de esta carpeta. En ella de entrada ya encontramos bastantes archivos clasificados en otras tantas carpetas. Tenemos por ejemplo el directorio "pages", donde se irán colocando las páginas (pantallas) de tu aplicación, así como los “assets” (archivos de imágenes y otro tipo de materiales externos que vas a usar en las páginas), o el componente app, que es la raíz de nuestra aplicación.

Encontrarás una serie de archivos sueltos, como el index.html, que es el punto de arranque de la aplicación web. Encontrarás también el archivo service-worker.js, que es el service worker (de las Progressive Web Apps) que usarás si deseas publicar tu proyecto de Ionic como una aplicación para la web. O el manifest.json, también para definir una Progressive Web App.

Carpeta www

En la carpeta "www" están los archivos que se producen cuando realiza toda la parte de la transpilación del TypeScript, el compilado de los archivos Sass, el linting del código, etc. En ese proceso, todos los archivos de "src", se transforman y se vuelcan en la carpeta "www". Esos archivos producidos en "www" son los archivos de una aplicación web, con Angular, que se podrán visualizar correctamente en el navegador.

Durante la mayor parte del proceso de desarrollo vamos trabajando con el proyecto de Ionic y vamos visualizando los resultados en el navegador. En resumen, en www están los archivos que se están visualizando desde el navegador al hacer el ionic serve, o los archivos que se tomarán cuando vayas a producir tu aplicación para cada plataforma.

En esta carpeta no deberías editar nada, puesto que los cambios se machacarán la próxima vez que se sirva el proyecto para visualizarse a través del navegador, o al hacer el *build*. Realmente tienes que editar lo que desees en la carpeta "src". En este sentido un error típico es editar cualquier archivo, como el index.html, de la carpeta www y cuando vas a visualizar los cambios encuentras que no se ha modificado nada. Es debido a que el índex.html de la carpeta "src" es el que se toma como punto de partida para producir el index.html de www y en la operación de build (o al hacer "ionic serve") tus cambios se habrán machacado.

Podrás observar que en la carpeta www no hay ningún archivo TypeScript (extensión ts), ya que se ha transpirado para producir los archivos JavaScript (.js), que realmente entiende el navegador. De manera parecida, los CSS ya no tendrán código Sass.

La carpeta "www" también tiene código minimizado y como se ha dicho, es la que se utiliza para construir los archivos finales que se llevarán al *store*. Lo que es interesante, porque si alguien a través del apk de Android quiere obtener los archivos fuente, lo único que encontrará será el código minimizado y ofuscado, y no el código que nosotros hemos usado a la hora de desarrollar. En versiones previas de Ionic 2 no existía esa dualidad de carpetas (src / www) y tampoco en Ionic 1, por lo que sí era posible obtener el código que habíamos usado como fuente para desarrollar.

Las siguientes carpetas también son importantes, aunque ya no serán usadas en tu trabajo diario de desarrollo.

Carpeta plugins

En esta carpeta se sitúan todos los *plugins* nativos con los que vamos a trabajar en la aplicación, ya sean de Cordova o del propio Ionic, o de terceros. De momento en un proyecto

recién iniciado podrás observar que ya se encuentran varios *plugins* básicos. Otros se irán instalando a medida que vayamos necesitando.

Carpeta platforms

En este directorio encontramos los archivos de cada plataforma a la que vamos a dar soporte en nuestra aplicación. Por ejemplo, si damos soporte a Android tendremos el subdirectorio "android" y si damos soporte a iOS encontraremos la carpeta "ios".

Directorio resources

La carpeta "resources" es importante a la hora de configurar iconos de la aplicación, o el icono de la pantalla splash (que es la imagen que aparece mientras tu aplicación se está cargando).

En esa carpeta tenemos que colocar iconos que deben cumplir unos requisitos, dado que se van a producir también internamente en diversos tamaños de manera automática por Ionic, para generar todas las variantes de iconos que necesita cada sistema operativo de destino. Soporta archivos en png o incluso psd.

Directorio hooks

Esta carpeta contiene una especie de disparadores que se ejecutarán ante determinadas situaciones. Generalmente vienen de Cordova y los puedes usar para la más variada gama de operaciones. Son como scripts que puedes crear y que se ejecutarán automáticamente al pasar cosas, por ejemplo, antes de compilar un proyecto, después, etc.

Ionic usa alguno de estos *hooks*, por ejemplo, al momento de compilar quita todo el código CSS que pertenece a otras de las plataformas. Esto lo hace para aligerar el proyecto, ya que a la hora de desarrollar Ionic produce el CSS de Android, iOS y Windows, pero no necesitas todo ese CSS si lo que estás produciendo es el apk para Android, por ejemplo.

Carpeta node_modules

Esta carpeta realmente se debería de conocer de otros proyectos para otros sistemas web o NodeJS. Son las dependencias de npm que vienen definidas en el package.json e instaladas en local dentro de tu proyecto.