En este documento se pretende recoger de una manera sencilla algunos aspectos del desarrollo de esta aplicación.

Versiones del sistema compatibles

A partir de la versión 2.3.3 (nivel api 10) hasta la última versión 4.3 (nivel api 18). Se pretende dar soporte al mayor porcentaje de dispositivos Android del mercado.

Según datos oficiales de Android (http://developer.android.com/about/dashboards/index.html#Platform):

Versión del sistema	Cobertura de mercado
2.3.3-2.3.7	33.0%
3.2	0.1%
4.0.3-4.0.4	22.5%
4.1.x	34%
4.2.x	6.5%
4.3	(sin datos)

Por lo que resulta en más de un 96% de cobertura.

Metodología de trabajo

Se ha seguido una metodología de trabajo iterativa e incremental, aplicando en la medida de lo posible la metodología Agile, desarrollando features en ciclos cortos, de forma incremental, obteniendo al final de cada iteración una aplicación estable y ejecutable (releases). Empezando por el proceso de login en la cuenta de Dropbox del usuario, después implementando la lista de libros, posteriormente la visualización de los detalles de un libro y por último las ordenaciones de las listas.

Librerías utilizadas

En el desarrollo de esta aplicación se han utilizado dos librerías, por un lado la librería que ofrece Dropbox para utilizar sus servicios (https://www.dropbox.com/developers/sync/sdks/android) y por otro lado una librería opensource para trabajar con epubs en Java (http://www.siegmann.nl/epublib).

Para trabajar con Dropbox se ha decidido utilizar la reciente **Sync API** frente a la Core API, las razones por las que se ha decidido trabajar con Sync en vez de Core, teniendo en cuenta el período de tiempo disponible para el desarrollo de la app, son básicamente por facilidad de aprendizaje y rapidez en el desarrollo.

Sync API abstrae al código cliente todo el trabajo de sincronización con la cuenta Dropbox con la que se está trabajando, liberando de su tratamiento al desarrollo cliente.

Otra diferencia importante entre Sync API y Core API es que utilizando Sync API el código cliente solo tiene acceso a una carpeta “Aplicaciones” -> “nombre_de_la_aplicación” dentro de la cuenta Dropbox del usuario, en lugar de tener acceso completo a toda la cuenta (es lo que Dropbox califica App folder permission, en lugar de Full Dropbox permission), pero como se ha mencionado anteriormente, esta librería ofrece la posibilidad de un desarrollo más rápido y sencillo. **Importante:** Esta es una cuestión importante puesto que para el correcto funcionamiento de la aplicación, los libros electrónicos del usuario tienen que estar alojados en dicha carpeta, para que la aplicación pueda acceder a ellos **(“Aplicaciones” -> “BookDrop” -> “ebooks”)**

Otra librería que se está utilizando en el proyecto es la librería opensource **Epublib** de Paul Siegmann para el tratamiento de libros en formato electrónico en Java.

Tras llevar a cabo una pequeña búsqueda y evaluación de distintas librerías de este tipo, el motivo que ha llevado a elegir esta es, de nuevo por las necesidades del proyecto, la facilidad de aprendizaje y uso.

Esta librería se utiliza básicamente para obtener epubs a partir de los archivos de Dropbox.

Algunos detalles de implementación

Se ha prestado especial atención al soporte a la mayor cantidad de dispositivos posible para alcanzar un mayor mercado y por lo tanto, un mayor número de usuarios. Es por ello que el primer objetivo era utilizar un **minSdkVersion** lo más bajo posible para alcanzar a la mayoría de dispositivos.

También se ha prestado atención al soporte a dispositivos con pantallas de gran tamaño (**tablets**), para que los usuarios de estos dispositivos tuvieran una buena experiencia de usuario. Es por ello que se ha decidido utilizar **Fragmentos** y definir un **layout específico** para tablets con dos Fragmentos visibles simultáneamente, uno con la lista de libros y otro con los detalles del libro seleccionado.

El uso de Fragmentos facilita las posibles ampliaciones futuras y la cobertura a un mayor tipo de dispositivos, utilizándolos de la forma que sea necesaria en distintos layouts definidos específicamente para cada tipo de dispositivo.

También se ha optado por dar soporte a dos lenguajes en la aplicación (inglés y español), para continuar con uno de los principales objetivos: llegar al mayor número de usuarios posible.

El sdk de Dropbox facilita en gran medida la interacción con el servicio de Dropbox, pero se ha necesitado un **Loader** para cargar **asíncronamente** el contenido de la carpeta de eBooks y así evitar errores ANR (Application Not Responding) que debiliten la experiencia de usuario.

Dificultades

- Conseguir una experiencia de usuario lo más fluída posible, cargando asíncronamente el contenido de la carpeta de Dropbox.

- Utilizar Fragmentos para utilizar layouts compatibles con distintos dispositivos.

Fe de errores y/o mejoras futuras

- La ordenación de eBooks por fecha de creación falla, necesita bugfix.
- La ordenación de eBooks por título ordena correctamente la lista, pero el resultado de la ordenación no se muestra inmediatamente en la lista, se debe cambiar de pantalla y volver a la lista para que se pueda ver ordenada (se piensa que está relacionado con el adaptador y algún problema en el notifyDataSetChanged() en el método sort() del ArrayAdapter que no está funcionando correctamente después de ordenar la fuente de datos del adaptador).

- No se contemplan todos los casos de sincronización de contenido con la cuenta de Dropbox, pudiendo resultar la app en un estado incoherente, se debe realizar un tratamiento más detallado del ciclo de vida en la interacción con el servicio Dropbox.

- No se tiene en cuenta que pueda haber carpetas dentro de la carpeta de trabajo de la app, es decir que dentro de Aplicaciones/BookDrop/ebooks haya subcarpetas con ebooks, solo se tienen en cuenta los ebooks en el primer nivel de la jerarquía de archivos a partir de ebooks/.

- Mejorar el tiempo de carga de ebooks. A pesar de que los contenidos estén cacheados localmente (según dice la doc del sdk de Dropbox) y que la descarga la lleve a cabo el sdk. Se ha intentado mejorar en la medida de lo posible este aspecto llevando a cabo una carga asíncrona a través de un Loader, pero se piensa que necesita una mayor velocidad.

- Hay datos sensibles hardcodeados en el archivo Util que deberían ir ofuscados o encriptados de alguna manera para evitar su uso no autorizado.

- No se ha prestado especial atención a la parte visual de la aplicación, debido principalmente a la falta de tiempo para llevar a cabo una buena experiencia de usuario, a pesar de ser un aspecto muy importante en el mundo de las apps.