Watchtower Publications Offline Search Tool

(JW PubSearch for Android)

Premise:

Jehovah's Witnesses are enjoying the benefits derived from electronic versions of literature provided by JW.org on their mobile devices. However, at the present time, a mobile device user has to be connected to the JW.org website in order to use search functionality to find information in the Watchtower Library. It is possible to download static file versions of literature for offline use, but their is currently no method to search through the text of a collection of downloaded documents for specific information. Also, to manually check for and download newly released document files is inconvenient.

Many cannot afford to subscribe to mobile data plans that allow for a constant Internet connection. Even for those that can, they do not want to exceed their bandwidth limitations and be charged overage fees. Therefore, many users typically use a WiFi/LAN based Internet connection whenever possible, especially to download large files. Then, when away from an Internet connection, they will have their data with them to use offline. Many people who download Watchtower publications from JW.org follow this use pattern.

There is a need for a mobile application that will automatically download new documents while online, and provide full text search and view capability of those documents during offline use.

Solution:

Create an application for mobile devices that allow users to automatically download, organize, search through, and view search results in Watchtower publications local to their device. Downloading publications and syncing index files require an Internet connection, but all other functionality shall be available offline.

There seems to be a trend for publications available on JW.org to be released first and foremost in the PDF format. With that in mind, the application will only support downloaded PDF files initially.

In order to provide this capability with a mobile application, three basic software components are needed. Since creating indexes is a processor and battery power intensive process, it is not practical for all three components to run on a mobile device. Therefore only *one* software component is proposed to be a mobile application, while the *other two* software components run on separate platforms. First, an indexer application will be developed to create indexes and sync them with a web service application. Next, a web service application will be developed to synchronize and distribute index files to all mobile users. Finally a mobile application will be developed such that end users can interact with their PDFs as described above.

Competition:

Currently, nothing matching what we propose exists on Google Play. This provides a unique opportunity to build something never before experienced by many mobile users, and by Jehovah's Witnesses as a group.

There is an iOS app called Equipd that caches HTML content from JW.org (including Watchtower Library Online) to iPads and iPhones for offline use. It does not download static files, and it does

not provide search capability for downloaded documents. However, it at least gives Jehovah's Witnesses a means to search and view text from Watchtower publications offline. More here: http://equipd.me/

With the above in mind, we propose to develop an Android application first, since Android users have no method to search through a collection of Watchtower publications offline. Many Jehovah's Witnesses use Android devices because they are economical, so this app will benefit a broad base of users. Also, there is a need for Android, Java, and C++ developers at Bethel, so developing for this platform will be good experience in developing skills that could be used elsewhere in Jehovah's earthly organization.

Features & Benefits:

Project:

- The main goal of the project is to satisfy a legitimate need for people we care about (Jehovah's Witnesses) - which *benefits them*. In the process, we will learn computer languages, development environments, and techniques that are needed by Jehovah's organization on supported platforms. This *benefits us* as developers.
- The application has the potential to be quite popular. We will hopefully get much user feedback and become embedded in a unique user community. This may result in future economic, spiritual, and/or other opportunities.

User experience:

- Automatically download and organize all PDF publications as they are released on JW.org.
- Full text search capability available offline for all downloaded PDF publications with a simple, familiar “search box” GUI component.
- View highlighted search results from the entire PDF collection in a contextual list.
- Search scope is at the “page” level. Each item in the list will represent keyword matches (hits) within the context of a “page” from an original PDF document.
- Clicking on items in the list will allow viewing of highlighted search results within the original print formatted publication with rich illustrations & graphics. This is provided for the first time ever for searchable Watchtower publications.
- Although search context is presented at the “page” level, the PDF viewer will allow viewing the original downloaded PDF document in its entirety.
- Simple configuration options allow users to customize their download directory, select auto-updating of index files, and configure the auto-download feature for PDF publications.

Efficiency:

- Unlike some mobile PDF indexing applications, index creation and updating will not be handled on a mobile device. Instead, a high powered remote PC takes care of the heavy lifting - thus saving much time, CPU load, and battery life on a mobile device.
- PDFs are an efficient format for storing graphically rich documents. Content is optimized, resulting in small file sizes.
- Index files are very small for the entire collection (less than 35MB at present).
- Index synchronization only transfers the changes in index files, not the entire index.

Therefore, syncing is more efficient and less resource intensive than processing.

- Apache Lucene is the backbone technology used for creating and querying the index. Lucene is known in the industry as fast and efficient.

Operational cost:

- Indexer application is run on low cost commodity PC hardware that requires a constant Internet connection.
- Web service application is a simple file sync manager run on an inexpensive hosted website.
- Open Source software tools are used where possible to speed up development time and reduce development effort.

Expandability:

- Lucene provides tokenizers and analyzers that support multiple human speech languages. It is possible to develop indexes for multiple languages in the future that will still use the same basic application(s). English will be the initial language, but configuration and support for multiple languages can be designed into the initial architecture.
- The PDF format is open, powerful, and has good developer support in the open source community. Expanded features like PDF printing, text/image exporting, document customizing, etc. may be explored in future versions.
- Support for more file formats, like ePub, could be explored.
- Modular design of application source code (classes) may make it easy to port the application(s) to other mobile and desktop platforms.

Education:

- Development of this project will afford the opportunity to get good training and experience with Eclipse, Java, Android SDK, Lucene, Solr, PDFBox, Lazarus(Pascal)/QT(C++), R-Sync, Web services, etc. These are all useful and relevant tools and technologies.
- Intellectual property created and techniques learned could be monetized later in similar commercial projects.
- Another finished “real world” software project looks good on our website and our resume's.

Monetization:

This application is intended to be released as free (gratis) software, so there is no monetization model needed. However, the lessons learned from research and from the development of proprietary code could be used on future commercial projects that follow a similar use pattern.

Software Architecture Overview:

Indexer application:

The indexer application is a software tool that will automate tasks related to creating, updating, and distributing an index. It will run on desktop PCs – Windows or Linux. It will be created with either Lazarus or QT Creator IDEs. The indexer application will initially work in conjunction with Apache Solr, a cross-platform Java application that uses the Lucene library for creating index files. Therefore, Solr will be a dependency at first. R-Sync will be a dependency in order to sync index files to the web service application. Ghost Script will also be a dependency, as it will be needed to

automate generation of thumbnail images from PDF files. The indexer application may take the form of a command line style utility, or it may have a GUI for monitoring or configuration purposes. Linux may be the preferred target platform at first, considering the dependency on R-sync. **NOTE:** some publications on JW.org will expire over time, so the index must react appropriately to that condition.

The indexer application has five major components, each of which performs the following tasks:

1. Checks for and downloads current PDF publications from JW.org and copies them to a pre-defined directory structure. It will monitor RSS feeds to trigger the downloading of magazines. A different strategy will be needed for automated downloading of other publications.
2. Splits the downloaded PDFs into single "page" PDFs to prepare for indexing as smaller "Lucene documents" (atoms).
3. Generates thumbnail images (using Ghost Script) for each PDF and stores them within in the same folder structure.
4. Generates index files by sending HTTP requests to Apache Solr server software. Solr is pre-configured to index Watchtower publications with the correct schema, tokenizers, and analyzers. Solr uses Apache Tika (built-in) to extract text from PDFs and format them into Lucene ready fields for indexing.
5. Synchronizes updated index files, thumbnail images, etc. with a web service application using R-sync. The web service application is hosted on a simple web server.

Web service application:

The web service application will run on a Linux web server hosted on the Internet. The web host must allow for shell access (SSH), support RSSH for password-less log-ins, provide and run R-sync, and allow clients to "push" and "pull" files from the web server using R-sync. The application may end up being as simple as a shell script that manages R-Sync connections with remote mobile users and the indexer application. It would be beneficial if asynchronous multi-threaded R-sync connections via a thread pool were possible.

A standard unlimited hosting Dreamhost account may be able to handle the requirements for a web service application at first, until download traffic becomes excessive. At that point, it may be prudent to provide a 256 MB memory VPS for the web service at $25.00 per month. A donation collection system could be put in place at this time to help cover the expense.

Android application:

The Android application will automatically download and organize all PDF publications to a mobile device as they are released on JW.org. It will provide the mobile user with full text search and viewing capability of all downloaded PDF publications with a simple GUI that consists of three screens.

GUI screens (Android activities):

1. The main search screen consists of a Title bar visible on the top of the screen, a *Search View* widget below the title bar, an *Expanded List View* below the *Search View* to display search results, a *Filter Menu* that will narrow search results down by various categories, and a *Sort Menu* that will control the order of search result items. Each component is described below:
    - The *Title Bar* will display a "Watchtower publications" style icon to the left of the application title - "Watchtower Publications Offline Search Tool".
    - The *Search View* widget will reside on an Android "action bar" and contain a search

input field with icon button on the far left side of the bar. It will enable users to type in search keywords and execute queries. The search input field will include a "search suggestions" feature that will auto-complete familiar terms as the user types. An Android "overflow menu" button will appear on the far right of the action bar, which allows access to additional screens, like the settings screen. An Android "action item" with appropriate icon for the Filter Menu will be to the left of the overflow menu. Pressing this button will open a Filter Menu. An Android "action item" with appropriate icon for the Sort Menu will be to the left of the filter action item. Pressing this button will open a Sort Menu. Since sorting and filtering is only applicable to returned search results, both action items will be inactive (grayed out) until a search query is executed.

- Below the Search View widget shall be an area to display search results in a tree-like *Expandable List View*. Multiple publications from the entire PDF collection will appear as separate items in a top-level list. Each item in the top-level list will represent a publication that contains search results in one or more of its pages. A thumbnail image of the publication will be shown with each top-level item. Publication list items can be clicked on and expanded to display child items that represent highlighted search results in context for *each page* that has them. Search scope is at the "page" level. Each item in the child list will represent keyword matches (hits) within the context of a "page" from an original PDF document. Child list items will be sorted either by relevance or page number, based on a user selectable setting in the settings screen.
- When the filter menu button is pressed, a *Filter Menu* will appear as a pop-up modal screen. The purpose of the screen is to allow the user to filter current top-level search results into smaller, more relevant lists using Android check-boxes. Lists items can be filtered by publication category (Awake, Bible, Booklets, Books, Brochures, Tracts, Watchtower, Yearbooks, and Other). Watchtower and Awake magazine items can be filtered by issue year and issue month. Watchtower magazine items can be filtered by edition type (Public, Simplified, Study).
- When the sort menu button is pressed, a *Sort Menu* will appear as a pop-up modal screen. The purpose of the screen is to enable the user to sort the order of top level list items using Android radio buttons. The user will be able to sort publications alphabetically (by title), by relevance, and by release date - in ascending or descending order.

2. A PDF viewer screen will open PDFs linked to search result items in the child list. Highlighted search results will appear within the downloaded PDF publication with rich illustrations & graphics. The PDF viewer screen will allow viewing the original downloaded PDF document in its entirety. Native, open source PDF viewers are available that can be embedded into the application, as opposed to coding a viewer from scratch. These provide many built-in features that will satisfy or exceed project requirements.
3. A settings screen based on Android's "Preferences" API will present configuration options that allow users to do the following:
    - Select or create a download directory using Android's Data Storage API (Internal Storage or External Storage, public or private).
    - Select either manual or auto-updating of index and thumbnail image files.
    - Configure the publication auto-download feature for PDF publications. The user

may select a subset of publications to download, or to manually download publications instead of automatic download.

- Select the sort order of child level list items (either by relevance or page number, ascending or descending).

In addition to the GUI, a file synchronize component based on Rsync and Android's Connectivity API will be implemented to automatically keep the local index files and thumbnail images up to date with the web service application.

Access to the index by the Android application will be provided by a custom Android Adapter class derived from the “Base Adapter” class. This way, all the search results can easily be presented in a List View. The Adapter will communicate with Lucene library classes to pass queries and handle results. The Lucene library is in native Java and relies on some base classes not existent in Android's Dalvik VM. A few tweaks, like deleting references to non-existent dependencies, will need to be done in order to use Lucene in an Android environment. An effort will be made to try to only import classes that are needed for querying Lucene (not indexing), thus keeping the source code smaller and more manageable.

Android's “Content Providers” API could be used to provide custom search suggestions derived from the index and stored in a separate “suggestions” SQLite data table. Or it could be used to derive search suggestions from a custom text file packaged with the application.

The user will be prompted at application start-up to download the latest publications and sync index/thumbnail files. Alternatively, an Android background service could be created to periodically check JW.org RSS feeds and other resources for changes and download new publications when they are detected. It could do the same for index and thumbnail files. The Android application will bind to the service and listen for events to send the user a toast when a new downloads are finished, etc.

Conclusion:

Where we go from here is totally up to us...