

# **Scripture App Builder**

## **Building Apps**

*Richard Margetts*

*Last updated: 23 January 2015*

# Contents

<b>1. How to build your first app</b> .....	<b>4</b>
<b>2. App Creation Basics</b> .....	<b>6</b>
2.1. <i>Does it matter what I choose for the package name?</i> .....	6
2.2. <i>Where do the default book names and abbreviations come from?</i> .....	6
2.3. <i>Do I have to create a new keystore for each app, or can I reuse the same keystore for several of my apps?</i> .....	7
2.4. <i>How can I include pictures in my apps?</i> .....	7
2.5. <i>How can I add a horizontal line to separate certain paragraphs in text?</i> .....	7
2.6. <i>I don't like the name "Scripture App". Have you thought of calling the app something else?</i> .....	7
2.7. <i>Can I build apps when I do not have internet access?</i> .....	8
2.8. <i>How do the glossary entry popups work?</i> .....	8
2.9. <i>Can I get the app to display non-Scripture books?</i> .....	8
<b>3. Fonts</b> .....	<b>9</b>
3.1. <i>What is 'Grandroid', referred to on the Fonts setup page?</i> .....	9
3.2. <i>When do I need to include the Grandroid libraries?</i> .....	9
<b>4. Audio</b> .....	<b>10</b>
4.1. <i>How can I associate audio files with the text?</i> .....	10
4.2. <i>How do I create the audio timing files for audio-text synchronisation?</i> .....	10
4.3. <i>How do I distribute the audio with the app?</i> .....	10
<b>5. Picture Story Books</b> .....	<b>13</b>
5.1. <i>What do picture story book files look like?</i> .....	13
5.2. <i>Where do the pictures go?</i> .....	15
5.3. <i>What audio timing labels are used?</i> .....	15
5.4. <i>Where can I get sample story book text and pictures?</i> .....	15
5.5. <i>What about recording audio?</i> .....	15
<b>6. Song Books</b> .....	<b>16</b>
<b>7. Distribution</b> .....	<b>17</b>
7.1. <i>How do I distribute the app to others?</i> .....	17

7.2. *I am trying to send the APK file by Bluetooth to someone but it is always unsuccessful. Other files such as music MP3 get sent ok. .... 18*

# 1. How to build your first app

To build your first app with Scripture App Builder:

1. Launch **Scripture App Builder** from its icon on the desktop.
2. Click **New App** on the toolbar. The New App wizard will appear.
3. On the first page of the wizard, specify:
  - the **App Name**, such as “Dogon Bible”, “Evangile de Luc”, “Yoruba NT”, etc. This is the main title of you app and will be seen by the user. Do not include underscores or hard to understand abbreviations.
  - the **Package Name**, a dot-separated string which uniquely identifies your app. For testing, try something like:  
`com.example.myapp1`  
`com.example.myapp2`, etc.
4. Click **Next** to move to the next page.
5. On the second page of the wizard, entitled **Books**, click **Add Books...** and select the Scripture books you want to see in the app. These can be USFM (Paratext) format, USX format, Digital Bible Library text release bundles, or a zipped file of USFM or USX files.
6. Click **Next** to move to the next page.
7. On the third page of the wizard, entitled **Fonts**, choose the font and text direction.
8. Click **Next** to move to the next page.
9. On the fourth page of the wizard, entitled **Colors and Features**, choose the colour scheme, book selector type and numeral type.
10. Click **Next** to move to the next page.
11. On the fifth page of the wizard, entitled **Signing**, you need to specify the keystore and alias to use to sign the app. An app must be signed in this way so that it can installed on an Android device.

If you do not already have a keystore file (which you are unlikely to have if this is your first time using the program):

- i. Click **Create New KeyStore Wizard**.

- ii. Enter a new filename for the keystore, such as “test” or something like that. Specify a password.
  - iii. Click **Next** to continue.
  - iv. Enter an alias name for a key to create within your new keystore, such as “testkey”. Specify a password and enter at least one piece of information in the fields below.
  - v. Click **Next** to continue.
  - vi. A new keystore will be created for you. Click **Close**.
12. Back on the **Signing** page of the New App wizard, you need to specify the keystore password, select the alias and enter the alias password (just as you entered them in the step above).
13. Click **Next** to continue.
14. Take a look at each of the app configuration pages by selecting them in the tree view on the left. Look in each of the tabs on each page to verify that you have the settings you want. You can always go back to them later to change them if you find you need to make modifications to fonts, colors, styles, etc.
15. When you have finished configuring the app, click on **Build App**.
- If something isn’t configured correctly for the build to work, you will be notified of this.
16. A black command box will appear. Wait about a minute while the app is compiled.
17. Copy the generated **.apk** file onto your phone or tablet and click on it to install it.

## 2. App Creation Basics

### 2.1. Does it matter what I choose for the package name?

Yes, those who install the app will be able to find its package name on their device. It will also appear in the web address for your app if you make it available on Google Play.

The standard is to begin with the reversed web address of the publishing organisation, e.g. if it is the Bible Society of Nigeria, the package name will begin with:

```
org.biblesociety-nigeria
```

and will be followed by something identifying the language and type of publication (New Testament, Bible, etc.), e.g.

```
org.biblesociety-nigeria.xyz.bible
```

If you work for a Bible translation or publishing organisation, you might have standards to follow for package names, so please contact your digital publications coordinator for advice on this.

Once you publish your app on an app store, you cannot change its package name later if you want users to continue to receive updates. The package name uniquely identifies the app in the Android world.

If you are building apps for **test purposes** on your devices, you can use a package name beginning with com.example, e.g.

```
com.example.test.app123
```

But remember to change it before you publish the app.

### 2.2. Where do the default book names and abbreviations come from?

The USFM standard is to use \toc2 for the book name and \toc3 for abbreviations:

e.g.

\toc1 The Gospel according to Matthew	(long book name)
\toc2 Matthew	(short book name)
\toc3 Mat	(abbreviation)

If Scripture App Builder does not find \toc2 or \toc3 it tries to guess the book name and abbreviation from \mt (main title) and \h (heading), but it is best to provide them in the toc fields.

For the Digital Bible Library text release bundles, the book names come from the metadata.xml file within the bundle.

You can change the book names and abbreviations on the book page for each book.

### **2.3. Do I have to create a new keystore for each app, or can I reuse the same keystore for several of my apps?**

You can use the same keystore and key alias for all or several of your apps.

See here for more details:

<http://developer.android.com/tools/publishing/app-signing.html>

### **2.4. How can I include pictures in my apps?**

You can include JPG or PNG image files in the text of the app.

The app recognises the USFM illustrations markers `\fig.... \fig*`:

`\fig DESC|FILE|SIZE|LOC|COPY|CAP|REF\fig*`

i.e. 7 parameters, separated by six vertical line characters.

The only required parameter is FILE – the filename. If the file extension is .tif, the app will look for a file with the same name but with a .png or .jpg extension instead.

You can optionally include captions (CAP) and chapter/verse references (REF).

Examples:

`\fig |picture1.jpg| || |This is a picture|5:3\fig*`

`\fig |picture2.tif| || |This is another picture|\fig*`

`\fig |picture3.jpg| || ||\fig*`

Include the image files in the app via the **Images** ➤ **Illustrations** page of Scripture App Builder. They will be included in the app assets and compiled into the final apk, so try to keep them as small as possible.

### **2.5. How can I add a horizontal line to separate certain paragraphs in text?**

Use the marker `\zhr` to add a horizontal line. This will add a `<hr>` element in the HTML.

Please note that this is non-standard USFM, so this marker should not be used if you are submitting this same file to the Digital Bible Library.

### **2.6. I don't like the name "Scripture App". Have you thought of calling the app something else?**

The program that allows you to define and build apps is called "Scripture App Builder" but the app itself doesn't have a name. It is up to you to choose the names for the apps you build.

You won't see 'Scripture App' anywhere in the apps you create, so feel free to use an appropriate name in an international, national or local language. App names that work well in certain contexts will be less helpful in other contexts, which is why SAB allows you to contextualise it.

## 2.7. Can I build apps when I do not have internet access?

The first time you build an app, you will need to be connected to the internet otherwise the compiler will fail. After that you can set the 'offline' version in **Settings** so you can work offline.

## 2.8. How do the glossary entry popups work?

To get glossary links working, do the following:

- 1.1. Add a glossary file as a book. This should have an id of GLO and keywords should be marked with `\k ..... \k*`
- 1.2. Mark words in the main text with `\w ..... \w*`. If the glossary form is different from the form of the word in the text, use the `|` character to specify the glossary form.

e.g. where 'angel' is in the glossary file:

`\v 5 You have made them a little lower than the \w angels|angel\w*` and crowned them with glory and honour.

## 2.9. Can I get the app to display non-Scripture books?

Yes, you just need to use standard format markers, like `\c` for chapter, `\p` for paragraphs and `\s` for sub-headings. The first marker in the file must be `\id XYZ`, where XYZ is a 3-letter code you choose (but don't choose a code already reserved for a Scripture book, e.g. GEN).

The book files must be **plain text files**. If you have Unicode characters, the text files should use **UTF-8 encoding**. To create a text file in Windows, use Notepad. To create a text file on a Mac, use TextEdit, remembering to choose Plain text files in the preferences because otherwise the default file type is RTF (which contains a lot of additional formatting codes).



## 3. Fonts

### 3.1. What is 'Grandroid', referred to on the Fonts setup page?

Grandroid (Graphite for Android) is a collection of native libraries from SIL Non-Roman Script Initiative (NRSI). They can be packaged within the app, enabling Android devices to make use of Graphite font rendering features.



Grandroid is not only about Graphite. It also fixes a few of the font display problems in recent versions of Android.

You can find more information about Graphite here:

[http://scripts.sil.org/cms/scripts/page.php?site\\_id=projects&item\\_id=graphite\\_home](http://scripts.sil.org/cms/scripts/page.php?site_id=projects&item_id=graphite_home)

You can find more information about Grandroid here:

<https://github.com/silnrsi/grandroid>

### 3.2. When do I need to include the Grandroid libraries?

This will depend on the font and special characters you need to display. The more complex your script, the more likely you are to need Grandroid support.

Please note that if a font displays correctly on your own phone without Grandroid, it does not mean it will display correctly on all phones and Android versions. As well as testing your app on the latest version of Android, it would be a good idea to test it on a phone running Android 4.2 or 4.3 (which have known font display problems) and the older 2.3 (which has limited built-in complex font support).

You will almost certainly need to include the Grandroid libraries:

- If you have a non-Roman script, e.g. Greek, Cyrillic, Armenian, Hebrew, Arabic, Syriac, Thaana, Devanagari, Grumukhi, Oriya, Tamil, Telugu, Kannada, Malayalam, Sinhala, Thai, Lao, Tibetan, Myanmar, Georgian, Hangeul, Ogham, Runic, Khmer, Ethiopic or Nko.
- If you have a Roman script which makes use of combining diacritics, such as separate acute accents or tone marks (e.g.  $\acute{u}$ , which is composed of two characters, but not  $\acute{e}$  which is a single character).

You are unlikely to need to use Grandroid:

- If you have a simple Roman script which does not make use of combining diacritics. So that means a-z, plus other IPA characters such as  $\epsilon$ ,  $\jmath$ ,  $\eta$ , etc. as long as they are not being combined with tone marks or accents.

If you try and display a complex script without Grandroid, you might find the following problems:

- Lack of Right-to-Left support - on Android 2.3 (Gingerbread).
- The system font being used rather than the font you specify - on Android 4.2 and 4.3 (Jelly Bean).
- Lines with combining diacritics being displayed in the system font, while other lines are being displayed correctly - on Android 4.2 and 4.3 (Jelly Bean).
- A blank screen where there should be text - on Android 4.2 and 4.3 (Jelly Bean).

If you want to display complex fonts, including Right-to-Left scripts, on Android 2.3 (Gingerbread), it helps if the font is Graphite-enabled since Android 2.3 has limited handling of OpenType for font rendering.

## 4. Audio

### 4.1. How can I associate audio files with the text?

You need to look in two places:

1. On the **Audio** tab for each book. Here you add the MP3 files and optional timing files for each chapter.
2. On the **Audio Distribution** tab for the book collection. Here you choose how you will distribute the audio (packaged inside the apk, in an external folder or via Internet download).

### 4.2. How do I create the audio timing files for audio-text synchronisation?

There is a separate document describing how to do this entitled “How to Generate Timing Information for Scripture App”.

Alternatively, if you are recording using HearThis, there is an export function which will generate the timing files for you. For more information about HearThis, see:

<http://hearthis.palaso.org>

### 4.3. How do I distribute the audio with the app?

There are 4 ways of including audio files in your app. Specify the method you want to use in the **Audio Distribution** tab of the book collection.

#### 1. Assets

The mp3 files will be packaged inside the apk file for the app. This is the easiest method for a few files (e.g. one book) and requires no permissions. But be beware that the apk will get very large if you have several books of audio. The maximum size of an apk that can be uploaded to the Google Play store is 50 MB.

## 2. External Folder

No audio files are packaged within the app, so the apk is small. The app will look in a specified SD card folder to find the audio mp3 files it needs. If you are distributing the app via SD card, you include the folder of audio files on the SD card together with the apk. This method requires the 'Read external storage' permission but not internet access.

You can place the mp3 files inside sub-folders and sub-sub-folders in the specified SD card folder, using any folder names you choose.

For example, here are some possible folder names:

```
\Bible Files
  \1 Old Testament
    \01 Genesis
    \02 Exodus
  \2 New Testament
    \01 Matthew
    \02 Mark
```

or

```
\NT Mamara
  \01 Macoo
  \02 Marika
  \03 Luka
  \04 Yohana
```

There is no need to use 01, 02, etc. in the folder names, but these can be useful if folders are listed in alphabetical order

Alternatively, you can place all the audio files in a single folder without using any sub folders.

## 3. Internet Download

Like method 2, no audio files are packaged within the app, so the apk is small. The app will look in a specified SD card folder to find the mp3 files it needs. If it doesn't find them there it can download them one by one when it needs them from a website of your choice. This method requires the 'Read external storage', 'Write external storage', 'Connection state' and 'Internet' permissions.

Recommended storage locations for the files on the internet include:

- Amazon S3 (Simple Storage Service): <http://aws.amazon.com/s3/>
- Google Cloud Storage: <https://cloud.google.com/storage/>

Once you have created an account with one of these cloud storage services, you create a 'bucket' in which to place your mp3 files. When you add the files, you need to make

them public and make a note of the web address link to use to access them, e.g. <http://s3.amazonaws.com/yourbucketname>

You will get some months of free storage before there is a charge according to the bandwidth used, i.e. how many MB of audio users download. It might be easiest to organise this kind of cloud storage at an organisational level rather than creating a new account for each language.

#### 4. FCBH Digital Bible Platform

If the audio files your app uses come from Faith Comes By Hearing, there is no need to upload them to the internet. They are already there and can be accessed via the Digital Bible Platform. You will need to register with FCBH and provide your app with the DBT key (received on registration) and volume id of the set of audio files. This method requires the 'Read external storage', 'Write external storage', 'Connection state' and 'Internet' permissions.

##### **Getting a DBT key**

The DBT key (the API Key) is a 32-character alpha-numeric string.

To get a key, you need to register with FCBH:

1. Go to <http://www.digitalbibleplatform.com>
2. Click **Sign Up** on the top right of the screen.
3. Click **Sign Up for Developer Access**.
4. Fill in your details, mentioning that you are building apps with *Scripture App Builder* and hit **Submit**.

##### **Getting the DAM ID**

When you get your DBT key from FCBH, you'll be able to search the library of volumes to get the DAM ID for the language audio you need.

The DAM ID is a 10-character string which uniquely identifies the set of audio files for a language, e.g. "MYKWBTN2DA" (for the Minyanka NT), "TPIPNGN2DA" (for the Tok Pisin NT).

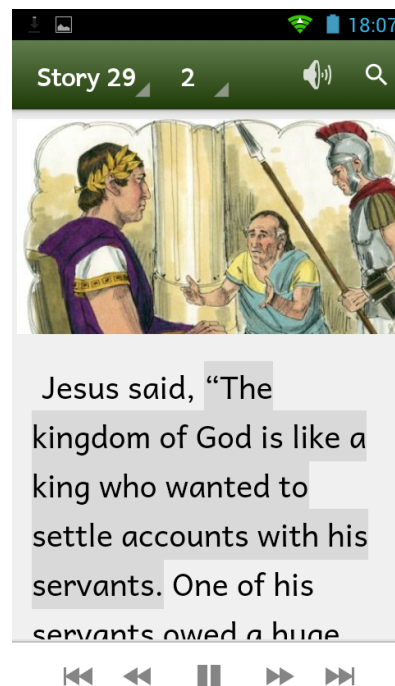
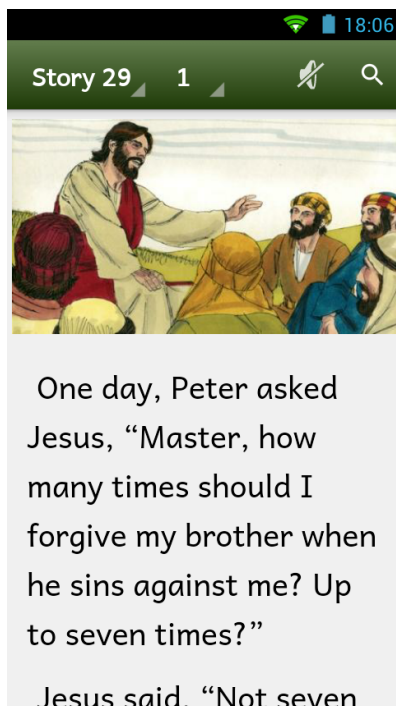
1. Go to the **Digital Bible Platform Developer Docs** website. <http://www.digitalbibleplatform.com/docs>
2. Go to the **Volume Listing** page. You can find this in the menu on the left of the screen, under **API Reference** ➤ **Library Catalog**.
3. Scroll down the page to the **API Explorer**.
4. Type your DBT key into **key (required)**:
5. Type the language code into **language\_code**.
6. Click the **Test** button.

- In the list of volume results, look for the **DAM ID** of the volume which has audio content (rather than text content):

```
"media": "audio"
```

## 5. Picture Story Books

A picture story book is a book with a picture at the top of each page and a few sentences of text underneath. In the app, the picture will stay at the top of the screen and the text will scroll.



You can associate a single audio file with the whole picture story book rather than an audio file per page. Audio-text highlighting works as for Scripture, and when the audio reaches the end of a page the app will scroll across slowly to start the next page of the story.

### 5.1. What do picture story book files look like?

They start with an id of your choice:

```
\id STO29
```

and a title:

```
\toc2 The Story of the Unmerciful Servant
```

Each page begins with the \page marker:

```
\page 1
```

followed by an image filename:

```
\img obs-29-01.jpg
```

and then the text for that page, using USFM markers like \p for paragraphs:

```
\p One day, Peter asked Jesus, "Master, how many times should I forgive my brother when he sins against me? Up to seven times?"  
\p Jesus said, "Not seven times, but seventy times seven!"  
\p By this, Jesus meant that we should always forgive. Then Jesus told this story.
```

Here is an example:

-----  
\id STO29

\toc2 The Story of the Unmerciful Servant

**\page 1**

```
\img obs-29-01.jpg
```

```
\p One day, Peter asked Jesus, "Master, how many times should I forgive my brother when he sins against me? Up to seven times?"
```

```
\p Jesus said, "Not seven times, but seventy times seven!"
```

```
\p By this, Jesus meant that we should always forgive. Then Jesus told this story.
```

**\page 2**

```
\img obs-29-02.jpg
```

```
\p Jesus said, "The kingdom of God is like a king who wanted to settle accounts with his servants. One of his servants owed a huge debt worth 200,000 years' wages."
```

**\page 3**

```
\img obs-29-03.jpg
```

```
\p "Since the servant could not pay the debt, the king said, 'Sell this man and his family as slaves to make payment on his debt.'"
```

**\page 4**

```
\img obs-29-04.jpg
```

```
\p "The servant fell on his knees before the king and said, 'Please be patient with me, and I will pay the full amount that I owe you.' The king felt pity for the servant, so he canceled all of his debt and let him go."
```

etc.

The book files must be **plain text files**. If you have Unicode characters, the text files should use **UTF-8 encoding**. To create a text file in Windows, use Notepad. To create a text file on a Mac, use TextEdit, remembering to choose Plain text files in the preferences because otherwise the default file type is RTF (which contains a lot of additional formatting codes).

## **5.2. Where do the pictures go?**

Add them in the **Images** ➤ **Illustrations** page in Scripture App Builder.

You can use either PNG or JPG files. Try and make them as small as possible without compromising image quality. This will keep your app size small and reduce page load time.

## **5.3. What audio timing labels are used?**

1a - 1st phrase on page 1

1b - 2nd phrase on page 1

1c - 3rd phrase on page 1

2a - 1st phrase on page 2

2b - 2nd phrase on page 2, etc.

## **5.4. Where can I get sample story book text and pictures?**

Have a look at Open Bible Stories here: <https://door43.org/en/obs>

## **5.5. What about recording audio?**

Use your favourite sound recording software, such as Audacity.

You'll need to decide how fast or slow you want the reader to read. In some contexts, it might be good for picture story books to be read fairly slowly.

## 6. Song Books

You can build a song book app, with one song per 'chapter', e.g.

```
\id BAT
\mt Batoli Betiw

\c 1
\s Ala Tanu! Aleluya!

\q An k'Ala tanu
\q Ala Denke ko la,
\q O sara ka kunun
\q Ka jenamaya.
\b
\q2 Ala Tanu! Aleluya!
\q2 Ala Tanu! Amiina!
\q2 Ala Tanu! Aleluya!
\q2 Yesu be na tun.
\b
\q An bee tununna
\q Dibi Fanga de la,
\q Bari Matigi nana,
\q A ye an bee tila.
\b
\q A y'an kunmabo
\q A y'an bee horonya,
\q A y'a ni to an kono
\q K'an dusu jeya.
\b
\q Yesu mana na,
\q A na ke Masa ye,
\q A na dipe bee mara,
\q Ka Sitane tipe.

\c 2
\s An Fa Ala, Dubatigi

\q An Fa Ala, Dubatigi,
\q An be barika da i ye,
\q Ka i tanu, ka i bato,
\q I ka kan ni bonya ye.
\b
\q2 Kuma duman ke ne da la,
\q2 Ka ne Kisibaa ko fo.
\q2 Aleluya! Aleluya!
\q2 Yesu ye ne kunmabo.
\b
\q Ne be Ala sira taama,
\q An Matigi barika la.
\q Ne y'a don ko Ala sonna
\q Ne ka taa sankolo la
```

The `\id` marker can be any unique marker you choose.

The `\c` markers are used for the song numbers, `\q` for poetic lines and `\q2` for more deeply embedded poetry lines (such as for a chorus). Use `\b` for blank lines between verses.

Audio can be added, with text-audio synchronization so that each line is highlighted as it is sung.








The songbook files must be **plain text files**. If you have Unicode characters, the text files should use **UTF-8 encoding**. To create a text file in Windows, use Notepad. To create a text file on a Mac, use TextEdit, remembering to choose Plain text files in the preferences because otherwise the default file type is RTF (which contains a lot of additional formatting codes).

## 7. Distribution

### 7.1. How do I distribute the app to others?

You can choose one or more of the following options:

	<p><b>microSD Cards</b></p>	<p>Copy the APK file onto microSD memory cards. People insert the card into their phone slot and click on the APK to install it.</p>
	<p><b>Bluetooth</b></p>	<p>Use Bluetooth to transfer the APK from your phone to another phone. Once people receive the APK file they click on it to install it.</p>
	<p><b>Google Play</b></p>	<p>Publish the app on the Google Play Store. For this you will need a Google Developer's account. You will need to supply a description of the app, a large icon and screenshots of the app in action. Once people have the app installed on their phone, they will receive automatic updates when you make changes to the app.</p>
	<p><b>Email</b></p>	<p>Email the APK file to others. Once people receive the APK file they click on it to install it.</p>
	<p><b>Web Download</b></p>	<p>Post the APK file on your website or upload it to a Dropbox folder and tell others it is there.</p>

**7.2. I am trying to send the APK file by Bluetooth to someone but it is always unsuccessful. Other files such as music MP3 get sent ok.**

Some Android devices block you from sending or receiving APK files. This behaviour has been observed on Google Nexus devices. This is analogous to some email systems stopping you from sending .exe or .zip files because of security concerns.

To get round this, you can rename the file extension to MP3, send it, and then change it back to APK before installing it.