Using Anki 2.x and SyncFromXML to learn the language data in your dictionary.

Do you have data that you would like to drill yourself on in a flashcard program? Is that data available as an XML file? If so, you may be able to use Anki and this addon to pull your data in. (This tool has only been tested with LIFT input files, but it is a generic solution that could work with other XML formats too.)

Assumption: you want to do your editing elsewhere and simply pull the latest data into Anki; or, you will be importing your data **one** time and editing only in Anki after that. (But it may not be worth putting the effort into a sync configuration just to use it as a one-time import.)

Contents:

Table of Contents

KEY CONCEPTS

Anki is a free flashcard program that is very popular for language learning, and it is very flexible. The default **note type (a.k.a. model**), Basic, has two simple fields, Front and Back. For each **note** (i.e. data record) that you add to a **deck**, Anki will generate a **flashcard** that shows you Front to prompt you to remember Back. If you also to practice in the reverse direction, you can easily add another flashcard template for prompting from Back to Front, and then Anki will automatically generate two cards that draw their data from that one note. Each card will have its own history, since one direction is typically easier than the other. Anki will drill you strategically (SRS), showing the easier cards less often.

Key terms in Anki: deck, note, note type (formerly "model"), flashcard, Spaced Repetition System.
For help: http://ankisrs.net/docs/manual.html

But if you want to track more pieces of data than that, it's quite easy to create a **custom note type** for your deck that has more fields. You can then design various different flashcards that take advantage of the different fields. You can even use multiple note types with a single deck. For example, **the provided Anki shared deck has one note type, DICT_LIFT, for storing dictionary entries (recommended), and another, DICT_LIFT_EX, for storing example sentences** (recommended if you often have multiple examples per entry). Their two different types of notes would have different fields and different flashcard templates, but they could all be placed in a single deck for convenient

drilling.

One difficulty with this approach is that pasting and maintaining your dictionary data in Anki manually becomes tedious and error-prone. This can be avoided by using the SyncFromXML plugin, a tool that does a **one-way sync** from one or more XML files into Anki 2.x (not version 1.x). This 'sync' is essentially an import that can be run repeatedly without creating duplicates, and without losing the flashcard histories that Anki maintains. For the data that it imports, it silently overwrites whatever is in Anki already, but without affecting data it doesn't care about (such as existing cards' histories).

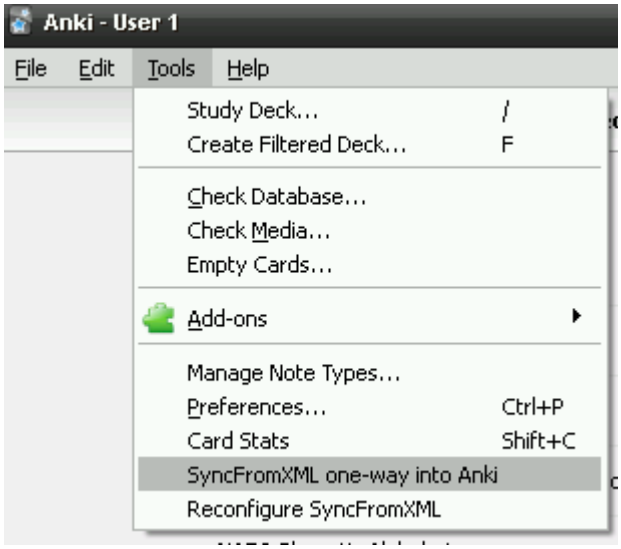
Warning: If you edit the field that is being used as the unique identifier of the record, it will no longer be considered the same record. The old Anki note and cards will be tagged for deletion, and the 'new' one will be imported from scratch, with brand-new cards.

The other main difficulty is that these custom note types in Anki are flat (non-hierarchical), so we cannot even try to fully duplicate the highly unpredictable tree structures supported by dictionary software such as FLEx or WeSay. However, for the purposes of language learning we can do well by copying the most relevant fields into a custom note type we feel is adequate, and by concatenating some repeating fields (such as sense-level fields like gloss) into single Anki fields. Other repeating fields, such as example sentence, might deserve their own Anki note type. Of course, all of this means that it would not be feasible to export data edits made in Anki back out to the dictionary software, and that's why SyncFromXML is strictly a one-way sync.

Steps to take (part 1):

0. Install Anki 2.x (here: http://ankisrs.net)
1. If you have an older version of SyncFromXML, back up your Documents\Anki folder.
2. Delete any note types / decks from previous versions that you don't need
3. Install SyncFromXML by going to Tools, Addons, Browse & Install, code 915685712. This information can also be found here: https://ankiweb.net/shared/info/915685712. Follow on-screen instructions--you may need to import the template types/deck by importing lift-dictionary.apkg (located in **documents\Anki\addons\syncxml\samples**).
4. You can try running the SyncFromXML addon now (Tools, One-way sync from XML) and either import the sample LIFT data or your own LIFT data. Or, read the next section first to understand what it will do.

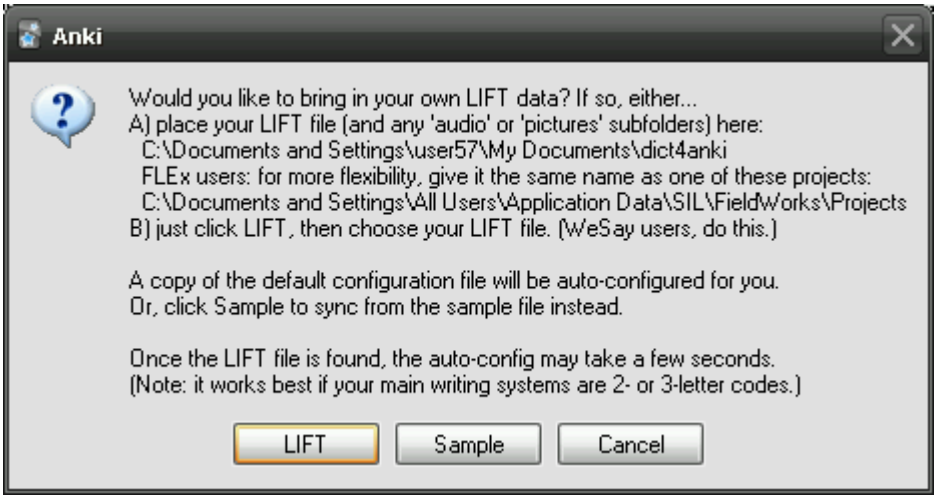
Auto-configuration wizard



If there is no file named SyncFromXML_config.txt in the syncxml folder (as is the case initially), the add-on will offer to auto-configure itself for a LIFT file. Typically this would be used with a LIFT file from WeSay or FieldWorks. (You can also import from the sample LIFT project just to see how it works, but then you'll want to delete those notes later.)

The fastest way to use this wizard is to initially place your LIFT file in a very specific location:
%userprofile%\documents\dict4anki
This is typically located somewhere like this:
C:\Users\Joe\Documents\anki4lift

It is also fine to just click the LIFT button and browse to find your file. (This is the correct choice for WeSay users. Remember, the sync tool will never modify that file in any way; it only reads from it, so it's safe for you to point to your actual data file.)



The resulting config file paths will generally look something like the following. (If you change the location later, you'll need to hand-edit all three paths in the config file.)
C:\Users\Joe\Documents\WeSay\Catalan

FLEx users will need to export (or send/receive) to produce a LIFT file before every sync operation in Anki. (It's ok to keep exporting overwriting the previous export.) It's easiest to just export to the dict4anki folder every time. It's also helpful to give the LIFT file the same name as your FLEx project's folder as explained below.

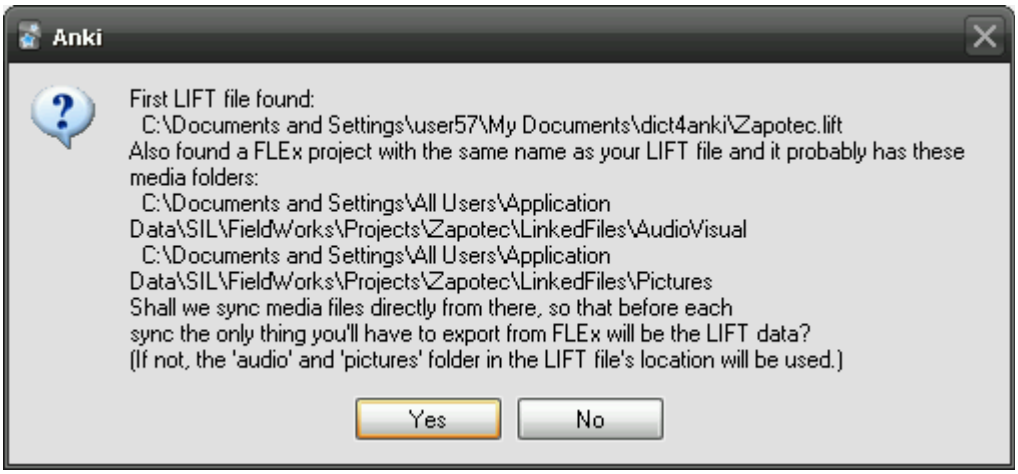
Anki flashcards support images and audio, and the one-way sync can copy media files from your WeSay or FLEx project into Anki. Once they are copied into Anki's collection.media folder, it should

be possible for Anki to sync it to a supported mobile device. (If, however, you'll just be using Anki on the same computer where your lexicon resides, you can save time by turning sync_media off in the config file: this will just reference your media files directly instead of copying them.)

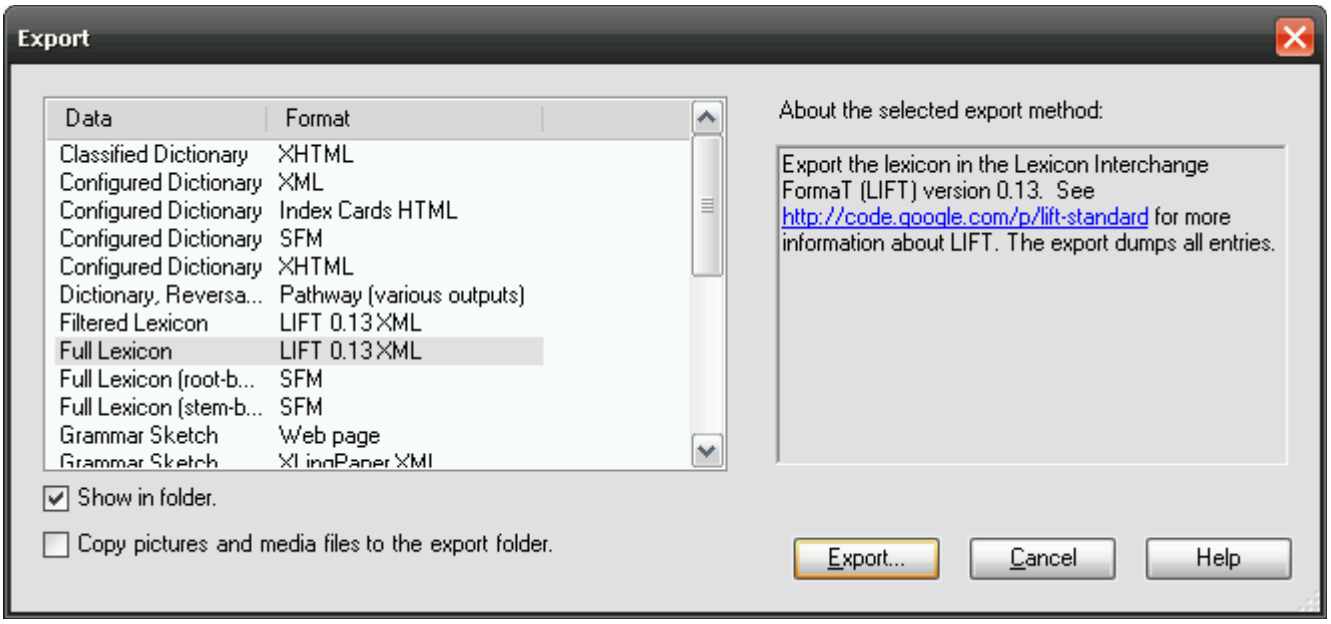
So, when you export from FLEx it is a good idea to give the LIFT file the same name as your FLEx project. This way, the wizard may then give you an extra option as the next step (provided your FLEx projects are in the standard location). For example, my username is 'user57', and I have a FLEx project named 'Zapotec', so if I export to Zapotec.lift then the wizard will notice this project folder:
C:\Documents and Settings\All Users\Application Data\SIL\FieldWorks\Projects\Zapotec

Note: on Windows Vista or newer, the standard path would look like this:
C:\ProgramData\SIL\FieldWorks\Projects\Zapotec

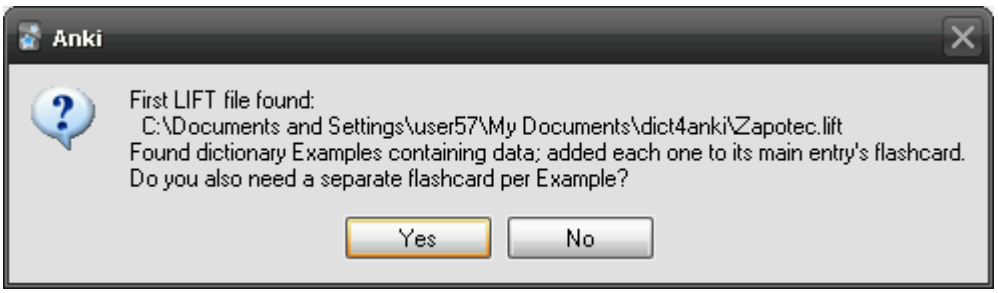
If I make sure I export my LIFT file as Zapotec.lift, then the auto-config wizard gives me this option:



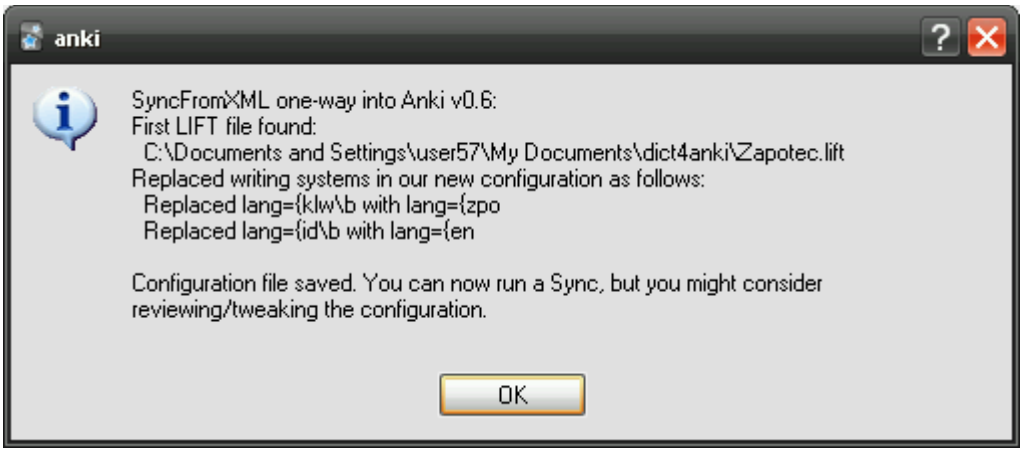
Using the LinkedFiles folders is probably the best choice if FLEx and Anki are only used on a single computer. This way, every time that you export to LIFT in order to sync, you won't need to tick the box to export your media files. The screenshot below shows the FLEx export dialog you'd use before each sync. Note that, if the FLEx project was detected, the second checkbox does ***not*** need to be ticked.



Once the file locations are clear, the wizard will try to analyze the LIFT file and determine your 'vernacular' and 'national' writing systems. It will start setting up the configuration file (SyncFromXML_config.txt) with instructions to sync each dictionary entry into one Anki note of type DICT_LIFT. If it detects any vernacular Examples in the lexicon, it will also configure a second "source" configuration to import each one to into one Anki note of type DICT_LIFT_EX. (See the introduction above.) But it will then ask you whether this is really needed. If you say No, the Example source configuration will still be saved but set as disabled.

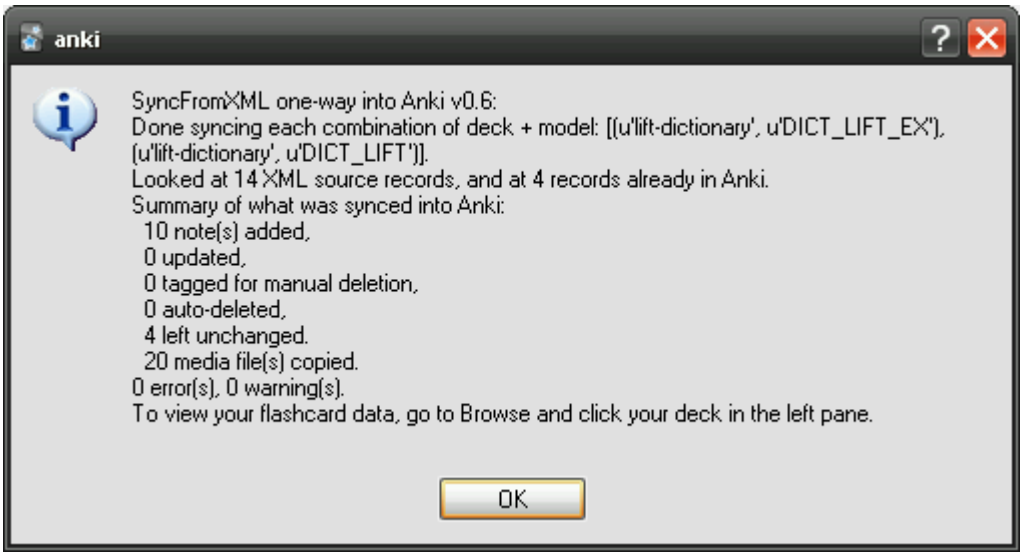


Once the auto-config wizard is finished, it gives a little report. It may also open the config file in Notepad so that you can view/tweak the resulting settings, or offer to "sync now".



The wizard initially copies the SyncFromXML_config_default.txt file which corresponds to the provided sample LIFT project, whose vernacular language is KLW, and whose national language is ID (Indonesian). In this case, it replaced KLW with ZPO, and ID with ENG, for both written and audio input system. (So, don't delete SyncFromXML_config_default.txt or you'll break the wizard.)

Next, try running the Sync command from the Tools menu to see how well this configuration will work. The sync will give a summary report of what it did (screenshot below).



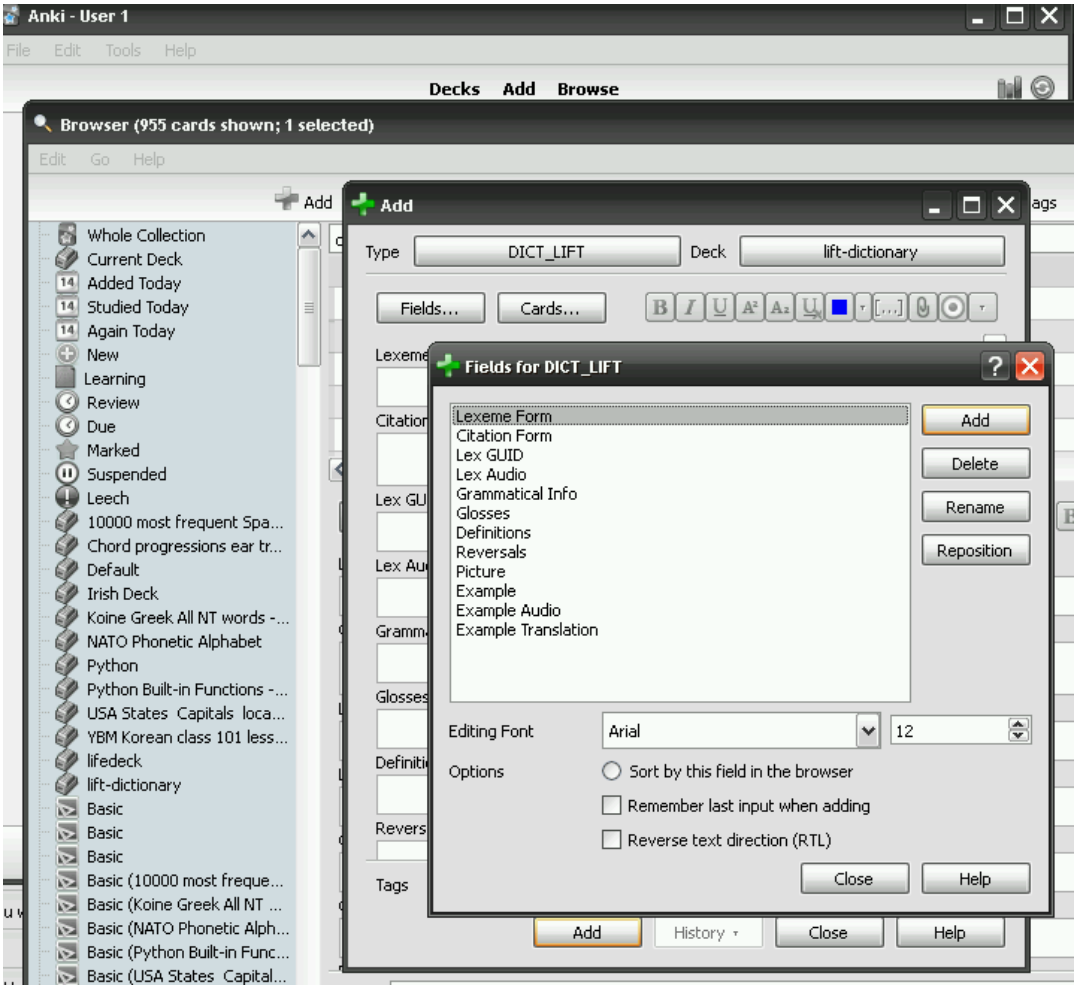
If there were errors or warnings, the log file (or its containing folder) may be opened automatically for your review. Also, try going into Anki's Browse view to review the imported data, then close the Browser and practice a bit with the flashcards to see what they look like.

That's it for initial setup! Click on "lift-dictionary" and "Study Now". To go beyond what the wizard was able to set up (or to import from some other XML format besides LIFT) you'll need to read the more detailed pages that follow.

NOTE TYPES (MODELS)

Initially, you'll probably want to just use the sample note types as provided by the add-on. Later on, you can optionally customize those types' fields and flashcard templates to meet your needs. But if you add/remove/rename any fields, you'll need to know how to change the settings in SyncFromXML's config file too. (That config file's main function is to map one or more source XML files to one or more "note type and deck" combinations in Anki.)

TIP: To customize a note type or its flashcard templates, click Brows, select a note or pretend to Add a new one, then select the note type you want (DICT_LIFT below), and click the "Fields..." or "Cards..." button. See the screenshot below.



The excellent Anki documentation, among other things, explains how to:
- design flashcards. (E.g. you can make a certain card auto-generate only if Glosses is non-empty.)
- format flashcards. (On-screen, that is. Anki's cards aren't printable unless you install another add-on.)
http://ankisrs.net/docs/manual.html

The fields provided in the template project, below on the left, correspond approximately to the data items listed on the right (as labeled in LIFT XML, a file format used by WeSay and FLEx).

Custom Anki field (DICT_LIFT note type*)	**LIFT label**
Lexeme Form	lexical-unit (vernacular writing system)
Citation Form	citation (vernacular)
Lex GUID	entry's guid attribute, which is an entry's unique ID
Lex Audio	lexical-unit (audio vernacular)
Glosses	gloss (national WS) - concatenate all glosses
Definitions	definition (national WS) - concatenate all definitions
Reversals	reversals (national WS) - concatenate all reversals
Picture	illustration (no WS) - grab first one only
Example	example (vernacular WS) - grab first one only
Example Audio	example (audio WS) - grab first one only
Example Translation	translation (national WS) - grab first one only

* Some other fields you may want to pull in using this note type: Category, Variants, Definitions. Also: some dictionaries have additional writing systems for Glosses, Definitions, and Example Translation.

A LIFT XML file contains much more than just the fields listed in the table above, but here is a trimmed example showing only those fields and where they fit in the structure of one entry:

```
<lift>
 <entry guid="81265940-eb8a-41dd-895a-21ad3e5511eb">
  <lexical-unit>
   <form lang="klw"><text>ata</text></form>
   <form lang="klw-Zxxx-x-audio"><text>634710572246718750ata.wav</text></form>
  </lexical-unit>
  <sense id="6b740f7c-0740-40a5-9ddc-0e1caa8cfb63">
   <grammatical-info value="Nomina (Kata benda)"></grammatical-info>
   <gloss lang="en"><text>roof</text></gloss>
   <gloss lang="id"><text>atap</text></gloss>
   <definition>
    <form lang="en"><text>roof</text></form>
    <form lang="id"><text>atap</text></form>
   </definition>
   <example>
    <form lang="klw"><text>Bajuku namenta nirata nu uda.</text></form>
    <form lang="klw-Zxxx-x-audio"><text>634710538956875000A LexExampleSentence.wav</text></form>
    <translation type="Free translation">
     <form lang="id"><text>Baju saya basah terkena hujan.</text></form>
    </translation>
   </example>
```

```
    <illustration href="atap.gif"></illustration>
   </sense>
  </entry>
</lift>
```

It would be nice to provide a configuration dialog or file where you could simply "map source fields A and B to target fields X and Y", but as you can see above, this is tricky because:
- We need to know how to find each XML element in the hierarchy.
- Some crucial 'fields' (i.e. guid) may actually be XML attributes, not elements
- The writing systems used will differ for each project
- Users' opinions as to which fields are important for flashcard drilling will vary, so it's hard to provide a 'stock' of settings. Still, it may be feasible to provide a config wizard for LIFT that does this.
- It would be nice to continue to support other XML formats besides LIFT.

The solution used by the SyncFromXML plugin is to dump all of the responsibility for this onto the configuration file, which is possible thanks to the magic of **XPath**, a standard syntax for specifying where to find something in an XML document. The xpath used for importing <entry> elements from any level in the document is simply:

```
//entry
```

Using that as a starting point, the xpath for then finding the Lexeme Form field's value for the "klw" writing system looks like this:

```
lexical-unit/form[@lang={klw}]/text
```

Likewise, the xpath for Lex GUID and Example Translation (Indonesian WS) are as follows:

```
@guid
sense/example/translation/form[@lang={id}]/text
```

By loading xpaths from the config file, the SyncFromXML addon itself doesn't need to know anything about LIFT, and you have the flexibility to pull in whatever you want to. Alas, this also means that you currently have to learn a bit of XPath syntax in order to tweak things, but hopefully the provided samples can serve as adequate recipes. The ones you don't need can be commented out, or vice versa. Note that removing or commenting out a field causes SyncFromXML to immediately start ignoring it (rather than clearing it out or updating it during the next sync).

The other sample note type provided is for handling example sentences, not lexical entries, so its starting point for finding all records is <example> rather than <entry>. Its xpath for finding the Example Translation data is therefore a bit shorter, even though it finds the exact same thing:

```
//example
translation/form[@lang={id}]/text
```

Here are the fields in the second sample note type. You might also want to include additional writing systems for Example Translations.

Custom Anki field (DICT_LIFT_EX note type)	**LIFT label**
Example	example (vernacular writing system)
Example Audio	example (audio WS)
Example Translations	translation (national WS) - concatenate all
Picture	illustration

The main difference here is that a different 'record' is in view than what we would think of as a 'dictionary record'. This is fine, but be aware that there will be no link in Anki between these examples and their entries. Also, no guid is provided by LIFT for examples (at present), so the example's text is used instead. This means that if the text in example changes in the LIFT file, then Example (and its flashcards) will be re-created from scratch.

Note: The two <source> elements above are pulling distinct data into distinct note types, but they're actually pointing to the same source file. Since SyncFromXML freshly reloads each source from the beginning of the file, it doesn't care. You can import any number of source files into any number of Anki decks.

Steps to take (part 2 of 3):

- If you haven't already, run "One-way sync from XML" from the Tools menu.
- Browse the collection and click the lift-dictionary deck to verify that the sample data imported correctly, and any old notes (the 2 placeholders from lift-dictionary.apkg) were tagged for deletion.
- Create a new deck with a name that matches your project's name somewhat (recommended). Or, if you want to just use this sample deck to contain your own data, delete all of its notes first.

NUTS AND BOLTS

Now let's look at how the one-way sync works and how to customize its settings. Again, the config file's main function is to describe a set of source XML records (and their fields) and map them to a corresponding note type (and its fields) in Anki. All of this is defined by one <source> element in the config file. (Multiple sources can be defined, representing multiple source files--or else, as we'll see in a moment, representing multiple passes through one source file.) Each <source> element targets exactly one note type (model) and one deck in Anki.

```
<source
  anki_deck="lift-dictionary" anki_model="DICT_LIFT" base_xpath="//entry" id_field="Lex GUID"
  source_file="samples/Lindu-Test.lift" source_audio_folder="samples/audio" source_image_folder="samples/pictures" >
```

```
  <source_field anki_field="Lex GUID" xpath="@guid" grab="first" />
  <source_field anki_field="Lexeme Form" xpath="lexical-unit/form[@lang={klw}]/text" grab="first" />
  <source_field anki_field="Lex Audio" xpath="lexical-unit/form[@lang={klw-Zxxx-x-audio}]/text" grab="first" is_audio="true" />
  ...
 </source>
 <source anki_deck="lift-dictionary" anki_model="DICT_LIFT_EX" base_xpath="//example" id_field="Example"
  source_file="samples/Lindu-Test.lift" source_audio_folder="samples/audio" source_image_folder="samples/pictures" >
  <source_field anki_field="Example" xpath="form[@lang={klw}]/text" grab="first" />
  <source_field anki_field="Example Audio" xpath="form[@lang={klw-Zxxx-x-audio}]/text" grab="first" is_audio="true" />
  <source_field anki_field="Example Translations" xpath="translation/form[@lang={id}]/text" grab="concat_all" />
 </source>
```

DELETIONS

For each combination of "deck and model" (i.e. note type) that you've targeted, any matching records in Anki that no longer correspond to any source record will be tagged for deletion. (To prevent this for a given flashcard, first move it to a different deck, manually.) In Anki's browse view, search for the syncxmlDel tag to delete these. (FUTURE: there may be an optional auto-delete option for these.)

Thus, if a record 'disappears' from the source XML file (because you deleted it, or because you edited its "ID" field), its old Anki note will be viewed as an orphan and tagged for deletion. You will be prompted to review the log file, which will list the problem records and suggest that you find and delete them. In Anki, go to Browse and search for tag:sfxdel or tag:sfxDel and delete the matching records.

MEDIA FILES

Option A. If you won't be using Anki's own sync feature for syncing your flashcards (e.g. to a mobile device), and if you're not planning to create a multimedia shared deck, then it's simplest not to copy your media files. Just set syncmedia="false" and the files will be used right where they are.

Option B. If you do set syncmedia="true", your media files will be copied and synced using much the same logic as with the XML data itself. Deletions are handled slightly differently, since files cannot be tagged for deletion in the same way, but you have two similar options:
- Look through the log file and manually delete the files it claims are orphaned.
- Use Anki's cleanup feature (applies to **all** decks): Tools, Maintenance, Unused Media .

There are three paths that you will need to specify in config in order to load your own data file. The first path, source_file, represents the XML source file. If any of the fields in that XML file contain links to media files, then the other two paths (which can be identical), represent where to find those media files. (These two base paths will be ignored for any links already stored as absolute paths.)

The default config file comes with paths that are relative to the addon's own location (which is something like D:\files\user7\documents\Anki\addons\syncxml):
   source_file="samples/Lindu-Test.lift"

```
source_audio_folder="samples/audio"
source_image_folder="samples/pictures" >
```

But assuming your own source file is located elsewhere, you'll want to specify absolute paths. Example:

```
source_file="D:\files\user7\documents\LIFT-from-FLEx\Lindu-Test\Lindu-Test.lift"
source_audio_folder="D:\files\user7\documents\LIFT-from-FLEx\Lindu-Test\audio"
source_image_folder="D:\files\user7\documents\LIFT-from-FLEx\Lindu-Test\pictures" >
```

(SyncFromXML tries to handle both kinds of slashes, but if you have trouble, try using backslashes if you're using Windows, or forward slashes for any other operating system.)

Duplicates: The XML file might point to files in multiple locations, and if so, some of them might have identical names. SyncFromXML detects such situations and reports them as errors so that you'll know which files to rename. (Note: Anki 2.x no longer allows any subfolders in the collection.media folder, so to avoid naming conflicts with other decks and to better detect outdated files, SyncFromXML prefixes the deck name to every filename.)

EDITING THE CONFIG FILE

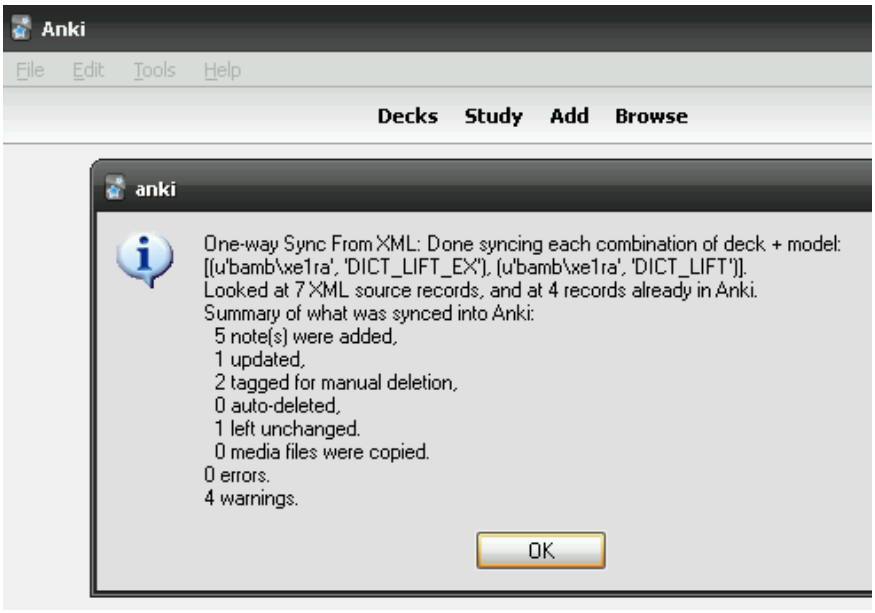
Hopefully that is enough information for you to successfully edit the config file (and the sample models) to meet your needs. (There is some more info about the config file in SyncFromXML_readme.txt.) For now, this task is a bit technical, but if this addon becomes popular, a graphical dialog for editing the configuration settings might be added.

Steps to take (part 3 of 3):

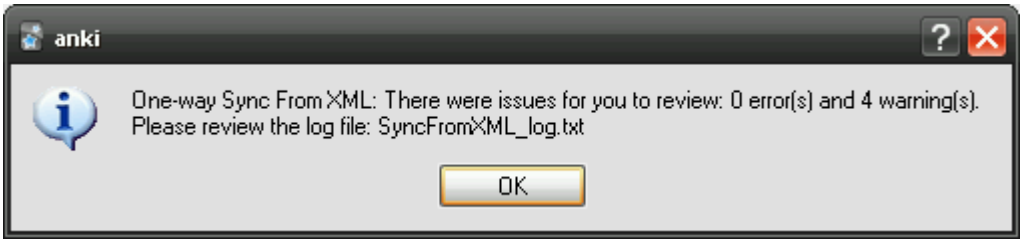
- Back up the config file, SyncFromXML_config.txt, and then edit it carefully.
- FLEx users: Since you're editing in FLEx rather than directly in the LIFT file, you'll want to export fresh to LIFT before every sync. Be sure to include media files, if your flashcards use them.
- Run "One-way sync from XML" from the Tools menu.
- Browse your deck and the collection.media folder to verify that the data and media imported correctly.
- Start drilling with your flashcards! Periodically update them by pulling from the XML file.

TROUBLESHOOTING

Logged errors: If SyncFromXML is able to run to completion, it will present a summary of what it did. Note that a deckname with special characters, such as bambára, may look funny on screen yet still sync successfully.



If there were any errors or warnings, it will suggest that you go read the log file (SyncFromXML_log.txt). If it crashes before that point, you won't see that suggestion, but you can still go open the file and look for clues. (Due to the crash, you'll want to first close Anki so that log file will be saved and closed.)



If the log doesn't quite pinpoint the problem, you can try closing Anki and editing the following line of code in logg.py from
VERBOSITY = VERBOSITY_NORMAL
to
VERBOSITY = VERBOSITY_DEBUG

If the cause of the problem is still unclear, you can send the log, your source XML, and your config file to the developer. (Tip: with unicode-related crashes, try VERBOSITY_SILENT temporarily.)

Missing records: Records will not be imported if their ID field is empty or non-unique, but with an ID they're imported invisbly if no cards can be generated (i.e. if the fields used on the flashcards' front sides are all empty). Workaround: place the ID field on the front side of a flash card template.