Consultant Level Training

These exercises are consultant level training in the use of the Field Linguist's Toolbox computer program. They mostly cover the underlying setup of the things that are done in the user level training, and that are already set up in the startup kit. Knowing the underlying setup can help you deal with problems or do customizations or adjustments when you want to.

You do not have to be a consultant to do these exercises. If you want, you can use them just to deepen your own understanding of Toolbox.

The Origin of these Consultant Level Exercises

These exercises are not a fully developed consultant training package. They are actually exercises that were included in the user training before the concept of startup kits was introduced. For many years all Shoebox and Toolbox training introduced setup from the ground up. Many users didn't do it themselves, but had a consultant help them get set up. Once the startup kits were built, the user training could be simplified a great deal. These exercises are the setup exercises that were removed at that time. One result is that these exercises do not attempt much detailed explanation. They walk through the necessary setup steps as quickly and simply as possible. As you do the exercises, you should read the help to get more detailed explanations of the steps in setup.

These exercises do not cover everything a Toolbox consultant needs to know. They only introduce the various topics. Nothing can substitute for experience using the program on a real or realistic project. If you aren't involved in an actual project, you can practice with almost any text corpus in almost any language. A grammar paper or book can be a fruitful source of lexical and morphological information about a language. It will include a number of annotated example sentences which you can enter as text and analyze. It may even include an appendix with a sample of connected text.

You should also read extensively in the help file. It has many very helpful articles on setup and use of the program. If you begin from the contents page, you will see the beginnings of many linked series of pages, each of which gives thorough coverage of a general area of usage of the program.

If you have access to a copy of the Shoebox User's Manual, either in hard copy, or as a doc file, you may want to read it. But the Toolbox help file is more thorough on most topics, as well as more accurate, since it describes Toolbox in all the places where Toolbox is different from Shoebox.

The Exercise Setting on a Project

One thing you should be aware of as a consultant is that exercise projects use a special exercise setting. This setting tells Toolbox that it should not overwrite or modify the original project files in any way. The result is that an exercise can be performed multiple times, and each time it starts up exactly the same as before. Toolbox holds everything in memory, so everything works the same as in a real project, except that no changes are ever saved to the exercise files. The user cannot force Toolbox to overwrite exercise files,

even with File, Save or File, Save All. In an exercise project, File, Save does not complain. It simply does nothing.

One of the results is that if you copy an exercise project to another folder and try to modify it for some other purpose, you won't be able to change it. But the exercise setting of a project can be toggled using an undocumented key sequence. Pressing Ctrl+Alt+Shift+T toggles the exercise setting of the current project. It gives a message saying whether the exercise setting has been turned on or off. Be careful of toggling this setting indiscriminately. If the exercise setting is being turned on, it performs a Save All, so toggling the exercise setting on and off can overwrite the original setup of an exercise project.

If you are teaching others, you can use the exercise setting to make custom exercises for your students. The basic approach is to set up the initial conditions of an exercises, then toggle the exercise setting on. Then test your exercise. If you need to modify any of the initial conditions, exit Toolbox, open the exercise project, modify the intial setup, and then toggle the exercise setting twice, off and then back on. Exercise projects can be used in a variety of ways for practice and homework. You would not, of course, use the exercise setting for homework in which the student turns in a modified form of the project to demonstrate that they actually did the homework.

The user level exercises all have the exercise setting turned on. But for various reasons, the consultant level exercises have it turned off. This means that as you do a consultant level exercise your changes will be saved. If you want to do one of them a second time, you should delete its folder and get a fresh copy of the initial setup from the original zipped file.

Exercise 1: Setup of Dictionary

This exercise will go through the steps in creating a new database type for a dictionary type of database, and creating a new dictionary file.

Open the exercise project.

Toolbox should show a completely empty box with no database window open at all.

Creating a Dictionary Database Type

In order to start a new type of file, we need to make a database type for it. The database type holds information about the possible markers in the file and lots of other things about the file. The list of available database types is under Project.

Choose Project, Database Types.

You will see a dialog box titled "Database Types". You want to add a new database type for your new dictionary.

Click Add.

You will see a dialog box that asks for the database type name and the record marker.

Enter "Dictionary" as the Name.

Enter "lx" (for lexeme) as the Record Marker.

Click OK.

You will see a large dialog box titled "Database Type Properties - Dictionary".

Click OK.

You will see the list of database types again. Now it shows "Dictionary". It also shows that "Dictionary" has a file name of "Dictiona.typ". This information will not usually be important, but it reflects the fact that the information for each database type is stored in a different file. A file that ends with ".typ" is a database type file.

Click "Close" on the list of database types.

Creating a Dictionary File

Now you are ready to create the dictionary file.

Choose File, New.

Navigate to the folder “Toolbox Training\4 Consultant Level\1 Setup of Dictionary”.

Enter a file name of "Amadup.dic".

Click Save.

It asks you to select a database type, with a default of "Dictionary".

Click OK.

You will see a new window titled "Amadup.dic". It contains nothing but the marker \lx.

This dictionary will be set up like the "Nawtulaikli" dictionary, with a typical entry having a lexeme, a part of speech, and a gloss.

Entering New Markers in the Dictionary Database Type

The first thing to do is to add the part of speech and gloss markers to the Dictionary database type.

Choose Database, Properties.

You will see a dialog box titled "Database Type Properties". It is showing a list of markers that contains only the "\lx" marker.

First you will enter the Part of Speech marker.

Click Add.

You will see a dialog box titled "Marker Properties".

Type "ps" as the marker.

Type "Part of Speech" as the Field Name.

Click OK.

Click OK on the Marker Properties box.

You will again see the empty dictionary entry that shows only \lx.

Insert an empty "\ps" field after the "\lx".

Choose View, Both Markers and Names.

You will see that \ps has a name, but \lx does not. We will add the name to \lx. The easy way to get to the properties of a marker is to do a right click on the marker.

Right click on \lx.

You will see a box titled “Marker Properties - \lx”.

Enter “Lexeme” as the Field Name.

Click OK.

The \lx field now shows a name of “Lexeme”.

One way to add a new marker to the database type is to type it into the marker selection box. We will add the \gl field in this way.

Choose Edit, Insert Field.

Type “gl” and press Enter.

You will see a box that says, “Marker not in marker list. Add it?”.

Choose Yes.

You will see the “Marker Properties” box for \gl.

Type “Gloss” as the Field Name.

Click OK.

You will see the dictionary entry with 3 empty fields, \lx, \ps, and \gl.

Making a Template

Since you will always want the \lx, \ps and \gl fields in every new dictionary entry, you can tell Toolbox to put them in automatically. This is done using something called a template. The first thing to do in preparing a template is to enter all the markers you want into a model entry that will be used as the template. We have a dictionary entry that contains three empty fields, \lx, \ps, and \gl.

Choose Database, Template.

You will see a dialog box saying that the template will be set to the fields in the current record.

Click OK.

Language Encoding

When you set up a new dictionary, you will usually want a language encoding for its language. The language encoding file contains information about the writing system and the sort sequence of the language. If possible, you will want to get a language encoding file that has been prepared by someone who has more experience with Toolbox. As a a consultant, you will want to collect language encodings for the languages used by the people you help.

In this example, you will set up a new language encoding for the Amadup language.

Choose Project, Language Encodings.

You will see a dialog box titled "Language Encodings" that shows the language encoding "Default".

Click Add.

You will see a dialog box titled "Language Encoding Properties".

Click on the Options tab.

Enter a Language Name of "Amadup".

It is nice to give a language a font with a distinct color, so we will do that.

Click Choose Font.

You will see a dialob box titled "Font".

Change the color to "Maroon".

Click OK for the font box.

Click OK for the language encoding properties box.

It will ask if you want to rename the default sort order from "added" to "Amadup".

Click Yes.

You will see the list of language encodings again. It now contains "Amadup".

Click Close.

Assigning a Language Encoding to a Marker

Now that the Amadup language encoding is available, you will tell Toolbox that the lexeme is in the Amadup language. This is done in the "Marker Properties" dialog box. A quick way to get there is to do a right click on the marker.

Do a right click on the "\lx" marker.

You will see the Marker Properties dialog box showing the properties of the \lx marker.

Choose a Language Encoding of "Amadup".

Click OK.

The Amadup dictionary is now set up and ready to use.

Exit Toolbox.

It asks if you want to save the file “Amadup.dic”. Toolbox does not ask this question on exit if “Auto Save” is turned on. It is recommended that users always have this option on. Toolbox defaults to having this option in any new project.

Choose Yes.

Look at the folder “1 Setup of Dictionary”. You will see that some new files have been added. The file “Dictiona.typ” is the database type file for the dictionary. The file “Amadup.lng” is the language encoding file for the Amadup language. And the file “Amadup.dic” is the dictionary file. As a consultant you should know the purpose of database type files and language encoding files. For example, to make a new language encoding available to a user, you copy its language encoding file into the user’s settings folder (the folder that contains the project file).

One thing that users should normally do differently from this setup is to put their data in a separate folder from their settings. This folder should be either over or under their settings folder. The Startup Kit has a “Data” folder under the settings folder. Another possible way to set up a user is to put the data in the main folder and have a “Settings” folder under it.

Exercise 2: Setup of Text for Analysis

This exercise will work through the steps required to set up a new text analysis project.

Open the exercise project.

Toolbox should show a window at the lower left titled “Amadup.dic”.

Importing a Text File

To begin this exercise, we will import a text file and set it up for analysis. The text file is named "Frog Meets Fish.txt". It contains a story text with a free translation of each sentence.

Choose File, Open. Navigate to "4 Consultant Level\1 Setup of Text for Analysis\Frog Meets Fish.txt" and click Open.

You will see a box titled "Import". It asks you to select an appropriate database type, or add a new one. Since this is the first interlinear text file we are importing, we need to add a new database type.

Click "Add a new Database Type".

You will see a box titled "Database Types". It shows "Dictionary" as the only available database type.

Click Add.

You will see a box titled "New Database Type".

Enter a name of "Text".

Enter a record marker of "id".

Click OK.

You will see a box titled "Database Type Properties - Text".

Click OK.

You will see the "Database Types" box again.

Click Close.

You will see the "Import" box again. Now it shows "Text" as an available database type.

Click OK.

You will see a window appear with the story of how Frog meets Fish.

Assigning a Language Encoding to a Marker

When a new database type is set up, we need to assign appropriate languages to the markers. In this case, the \txt line is in the Amadup language.

Right click on a "\txt" marker.

You will see a dialog box titled "Marker Properties - \txt".

Select a language encoding of "Amadup".

This is also a good time to fill in a field name for this marker. It has no field name because it was created automatically during the import of the text.

Enter a field name of "Text".

Click OK.

You will see all the text lines change to maroon color, which is the default color for Amadup language fields.

Adding References to the Text

The first step in analyzing a text is to add a reference before each sentence.

To prepare for referencing, we put a reference marker and a text abbreviation before the first text line.

Choose Edit, Insert Field.

Type "ref" and press Enter.

You will see a dialog box that asks if you want to add the marker to the marker list.

Click Yes.

You will see a dialog box titled Marker Properties - New Marker.

Enter a Field Name of “Reference”.

Click OK.

You will see the text window again.

The next step is to enter a text abbreviation into the reference field. This will be used in making the references and for concordances and word lists. It is best to keep the text abbreviation short, no more than 4 or 5 characters so that it does not take much space in a list of references. We will use "Frog" as the text abbreviation for this text.

Type "Frog" into the \ref field. (Do not press Enter.)

Next we are ready to add all the references to the text.

Choose Tools, Break/Number Text.

You will see a box titled "Set Up Numbering".

Click OK.

You will see a box titled "Database Type Properties - Text". It shows a page titled "Numbering".

Select "ref" as the reference marker.

Select "txt" as the text marker.

Click OK.

You will see a box titled "Break and Number Text".

Because this text is already broken into sentences, we remove the text break punctuation.

Delete everything from "Text-break Punctuation".

We entered the text abbreviation into the \ref field, so we will tell it to use that as the text name.

Under "Name of Text" select "ref" instead of "id".

Click OK.

You will see that the text now has a reference of the form "Frog.001" before each sentence. This reference is made up of the text abbreviation, a period, and a 3 digit number.

Changing the Size of the Text Window

It is useful to make an interlinear text window as wide as possible.

Change the size of the "Frog Meets Fish.int" window so that it is as wide as possible. Move it up as far as possible. Make it shorter so that it covers about the top half of the space available.

At this point, the "Frog Meets Fish.int" text is prepared for analysis.

Exit Toolbox.

Exercise 3: Setup of Interlinear

This exercise will work through the steps required to set up interlinearization for a text analysis project.

Open the exercise project.

Toolbox should show two windows, one titled "Frog Meets Fish.int", and one titled “Amadup.dic”, with a blank dictionary entry.

Setting up Interlinearization

Interlinearization refers to the process of breaking words into morphemes and putting glosses and parts of speech under the morphemes. The interlinearization setup for a text file is in its database properties. The focus should be on the interlinear text window.

Choose Database, Properties.

You will see the Database Properties dialog box. It is showing the list of markers used in the text file.

Choose the "Interlinear" tab.

Choose "Quick Setup".

You will see a dialog box that lets you set the markers for interlinear text.

Change the Text Marker to "txt".

Click OK.

You will see a dialog box that lets you set the dictionary file and markers to be used for interlinearization.

Select "Amadup.dic" and click "Insert".

Click OK.

You will see the Database Properties dialog box again. It now contains a list of three interlinear processes. The focus is on the first one, which is a "Parse" process, which means it breaks words into morphemes. We will make one change to the parse process.

Click Modify.

Turn on the check box that says "Output root guess".

Click OK on 2 dialog boxes.

Interlinearization is now set up and ready to use.

Click the Interlinearize button. (This is the white button just to the right of the 4 buttons with arrows.)

You will see the first sentence interlinearized. Three new lines have been added under the text line. In the \mb line each word appears with an asterisk in front of it to indicate that it has failed to parse. In the \gl and \ps lines each word has three asterisks under it to indicate failure.

Setting the Language Encoding

Since the \mb line is in the Amadup language, it should be set to the Amadup language encoding.

Right click on the \mb marker on the left side of the text window.

You will see a dialog box titled "Marker Properties".

Set the Language Encoding to "Amadup".

Click OK.

Exit Toolbox.

Exercise 4: Setup of Text Corpus

This exercise will work through the steps required to set up a text corpus and make a word list.

Open the exercise project.

Toolbox should show two windows, one titled "Frog Meets Fish.txt", and one titled “Amadup.dic”, with a blank dictionary entry. The first sentence of the upper window is interlinearized with all failure marks.

Setting up a Text Corpus

The best way to begin analysis is to use a concordance and wordlists. To do that, you must set up a text corpus. A text corpus is a list of text files with some information about what markers are used in the files.

Setting up a text corpus is done in the wordlist or concordance dialog box.

Choose Tools, Word List.

You will see a box titled "Create Word List".

Click Edit, to the right of "Text Corpus".

You will see a box titled "Text Corpora".

Click Add.

You will see a box titled "Text Corpus Properties". You need to enter a variety of information about the your text corpus.

Enter a Corpus Name of "Amadup Texts".

Select a Language Encoding of "Amadup".

Click "Edit Files List".

Select "Frog Meets Fish.txt", and click Insert to move it to the right column.

Click OK.

Change Markers to Process to "\txt".

Under Reference Markers, change Primary to "\ref".

Delete the Secondary and Tertiary so that they are empty.

Click OK.

Click "Close" on the Text Corpora dialog box.

You will see the "Create Word List" box, with the "Amadup Texts" corpus selected.

Making a Word List

Click "Create" to create the word list.

You will see a wordlist showing all the Amadup words in the Frog Meets Fish file.

Resize the wordlist window so that it is short enough to fit between the interlinear text window and the dictionary window. Shift it to the left side.

Sorting a Word List from the Right

It is also very useful to look at a wordlist sorted from the ends of the words. This can show patterns of possible suffixes.

Choose Window, Duplicate to duplicate the wordlist window.

Resize the second wordlist window to about the same as the first. Shift it to the right side.

Change the sorting of the second wordlist window to sort the words from the end. (Choose Database, Sorting. Turn on "Sort first field from end".)

Making a Word List of the Free Translations

One way to begin analysis is to identify possible words or roots, based on the free translation. Sometimes a comparative wordlist of the free translation can help with this process, so we will make one. To do that, we make another text corpus that looks at the same files, but looks at the free translation field instead of the text field.

Choose Tools, Wordlist.

Click Edit.

You will see the Text Corpora dialog box.

We can save some work by modifying a copy of the Amadus Texts corpus instead of making a completely new one.

Click Copy.

Change the Corpus Name to "Amadup Free Translation".

Change the Language Encoding to "Default".

Change the Markers to Process to "\ft".

Click OK.

Click Close on the Text Corpora box.

You now see the "Create Word List" dialog box, with the "Amadup Free Translation" corpus selected.

You must use a different file name for the English word list.

Change the "Output File" to "wordlistft.db".

Click Create.

You will see a wordlist of all the English words from the free translations of the sentences.

Make the window smaller so that is is only about a quarter of the total height of the main window. Put it in the lower right corner.

Exit Toolbox.

Exercise 5: Setup of Jump Path

This exercise will set up a jump path and show how to use it to insert new words into the dictionary.

Open the exercise project.

Toolbox should show five windows, one titled "Frog Meets Fish.int", one titled “Amadup.dic”, and three word list windows.

Setting up a Jump Path

In order to insert new words in the dictionary, we need to tell Toolbox how to jump from the \mb line of the interlinear text to the dictionary. That is done in the Database Type Properties.

Choose Database, Properties.

Click on the "Jump Path" tab.

Click Add.

You will see a box titled "Jump Path Properties".

Enter a Jump Path Name of "Amadup Dictionary".

Under Available Fields, click on "mb" and click Add to move it to the Fields to Jump From.

Under Available Databases, select "Amadup.dic" and click First to move it to Databases in Path.

Click OK on 2 dialog boxes to return to the text window.

The jump path is now ready to use.

To test this jump path, right click on "lyfch" in the \mb line of the text.

You should see a box titled “No Matches” with “lyfch” in an edit control.

Choose Insert.

You will see a new entry for "lyfch" appear in the Amadup.dic window.

Notice that this jump path causes jump from the \mb line to do operate differently than jump from the \txt line. Note that the word list windows did not jump to "lyfch". They jump from the \txt line.

Right click on "lyfch" in the \txt line of the text.

You will see the two word lists jump to show "lyfch".

This is a multiple jump that makes two windows jump at the same time. That kind of jump works only from a field which does not have a jump path. But if there is no jump path from a field, then it is not possible to make a jump insert into a dictionary. For this reason, a good setup is one in which the text line has no jump path so it can jump to the wordlist windows, and the morpheme breaks line has a jump path so it can jump to the lexicon and do insertions there.

Note that each window has a “Jump Target” setting under “View”. If a window has “Jump Target” turned off, then no jump will go to it. This can be used to prevent a particular window from moving in response to jumps.

Put the focus on the second wordlist window.

Choose View, Jump Target.

The check mark turns off.

Right click on “velgow” in the \txt line.

Only the first wordlist window jumps.

Parallel Jump

The jump target setting also prevents window tracking from parallel jump. Parallel jump causes two or more windows to track each other. For example, the first wordlist window will follow the second window.

Click on a word in the second wordlist window.

Both wordlist windows jump to show the word.

Press up arrow.

Both wordlist windows move together.

Click on a word in the first wordlist window.

The second wordlist window does not move to the word because it is not a jump target.

Press up arrow.

The second wordlist window does not move because it is not a jump target.

Put the focus on the second wordlist window.

Choose View, Jump Target.

The check mark turns on.

Click on a word in the first wordlist window.

The second wordlist window moves to show the word.

Exit Toolbox.

Exercise 6: Teaching Toolbox

The Teaching Toolbox folder contains both a short and a long demo. The short demo can be done in 10 minutes or less if desired, or can be extended some with additional discussion. The long demo can take from one hour to two hours, depending on how much discussion and commentary is added. These demos are useful for introducing the program, but are not sufficient to teach students how to use it.

For actual teaching, I prefer an extended approach using the full Toolbox Training. (This is the Dictionary, Text Analysis, and Starting Your Own Project modules, but not Consultant Level.) One can use the following sequence:

1. Introduce the subject of an exercise and demonstrate the exercise in class with a projector. Allow time for questions and discussion. This gives the students an overview of the subject, and gives them an idea what to expect when they do it.
2. Have the students do the exercise as homework. This reinforces their knowledge by letting them experience the actual keystrokes and mouse actions that are required to do the exercise.
3. After all the exercises have been completed, have the students set up a practice project with the startup kit. Give them a small amount of sample machine-readable text with analysis shown on paper. Have them enter morphemes into the dictionary and analyze the text so as to produce the correct analysis. This gives them time to become comfortable with the program before they become involved in the complexities of analysis.
4. Have the students set up a real analysis project with the startup kit. If they are working with live language speakers, have them enter and analyze new text that they elicit. If they are not working with live language speakers, then provide them with some other form of raw data to analyze.

You may want to help the students by setting up a version of the starter kit that has the language encoding they need already set up. For example, if the students will be using IPA, then an IPA starter kit can help the students get started more smoothly.

I take the view that the students should as soon as possible have a positive and successful experience doing some actual work with the program. Once that has happened, some will be motivated to learn more details, and will want to do the consultant level training, to read topics in the help file, and possibly to read other reference materials. But most will be satisfied to use the program successfully, and will not want to dig any deeper except when they have trouble getting it to do what they want.

Your students will move a little more smoothly through the exercises if you print out the instructions for the exercises and give each one a copy. This resolves some of the problems of working from an instructions file on the screen, such as getting the focus back on the right program after scrolling the instructions window. It also gives them a place to make notes for future reference.

On how long it takes to teach Toolbox, I estimate that the above process could take anywhere from 6 to 15 hours of class time. Much will depend on how comfortable the students are with computers and linguistics before they begin. Typically one can demonstrate 2 or 3 exercises in each class session, and assign the same as homework. This spreads the exercise phase over 5 or 6 class hours. The practice project time can be longer or shorter depending on how fast the students demonstrate their readiness to do real work with the program. As they start to do real work with Toolbox, some class time will be required to help them over their initial problems and misunderstandings.