Getting started with Emacs/XML

Lou Burnard

July 2001

1. Using Emacs to edit XML

GNU Emacs, in its SGML/XML mode, is a very attractive option for those wishing to produce high quality XML-encoded documents on a limited (or nonexistent) budget. The software itself is genuinely ubiquitous, very widely understood, documented in great detail, and comes at a price that every academic project can afford (i.e. zero). It may not be as flashy as some XML editors, but it can still be configured to make entering valid XML text very simple. In this exercise, we'll use a version that has been customised to simplify data entry of TEI conformant documents of any kind. We won't try to explain all about emacs as there are dozens of books and web pages on the subject (as a taster, there is a very good tutorial specifically on the XML customization of emacs), and emacs also comes with its own built in help system and tutorial.

2. Editing an existing file with emacs

Start Emacs by clicking on the Emacs icon. On the first screen that appears you will see the following mysterious suggestion:

```
If you want to create a file, visit that file with
    C-x C-f, then enter the text in that file's own  buffer.
```

Let's first explain the notation: **C-x C-f** means

1. hold down the CTRL key
2. without releasing the CTRL key, hit the X key once
3. release both keys and
4. hold down the CTRL key (again)
5. without releasing the CTRL key, hit the F key once
6. release both keys

It's probably a good idea to practice this manoeuvre, if you're not used to using the keyboard. Emacs dates from pre-mouse days and some of its delights are accessible only via the keyboard. Most of the commands available on the menus at the top of the screen are also accessible using the keyboard, and often once you've learned the keyboard combination you'll find it quicker to use that than to fiddle around with the mouse.

Whether you are going to create a new file, or open an existing one, you proceed in the same way, either by selecting "Open File" from the Files menu, or by typing C-x C-f. Whichever method you choose, the same (rather confusing) thing will happen: the cursor will disappear from the screen window and move to the bottom of the screen into a tiny window which is called in emacs-speak the *minibuffer*

The minibuffer is where you give emacs commands that require keyboard input. Emacs insists that you supply a name (even for a newly-created file), which is why the cursor has moved to the minibuffer. You can use the arrow or backspace keys to edit the text emacs is suggesting, and you can type in whatever you like – but you must type something. For this part of the tutorial, please enter xml/ballads.xml, and then press RETURN (if you get into a mess and want to start again, you can bail out of almost any emacs command by typing C-g).

Emacs now opens a new window, and you will see that the list of options available across the top of the screen has changed to indicate that you are now editing an XML file. If you've never used emacs before, you'll be relieved to learn that most of the keys behave in the way you'd expect (try moving around the screen with the arrow keys, PgUp Home, backspace, etc.). The mouse can also be used in the usual way to select text, scroll up and down, etc. Selected text can be deleted with the Del key, and most basic editing functions are available from the Edit menu. You may like to experiment with some of the commands there before proceeding. Note that a few key combinations don't do what you expect: C-c and C-v don't cut or paste because they are used by emacs for other purposes.

The file you are looking at contains just the text part of an extract from a putative electronic edition of a collection of Broadside Ballads (if you're interested, we took it from the Bodleian's wonderful collection at

http://www.bodley.ox.ac.uk/ballads). In this part of the exercise, all you have to do is to make this document *well-formed*. We have done most of the work for you: as you scroll through the text you might like to consider whether or not we have done it properly.

From the XML menu (6th from left), choose Validate. Or type C-c C-v. This will run the nsgmls XML parser to check the file you are editing. The command line used appears in the minibuffer at the bottom of the screen: press RETURN to confirm. In order to show you the output from the parser, emacs will split the screen in two. You should see the following messages in the new window:

```
nsgmls:ballads.xml:1:0:E: no document type declaration; will parse without validation
nsgmls:ballads.xml:61:50:E: general entity "c." not defined and no default entity
nsgmls:ballads.xml:61:52:W: reference not terminated by refc delimiter
nsgmls:ballads.xml:69:52:W: reference not terminated by refc delimiter
nsgmls:ballads.xml:77:52:W: reference not terminated by refc delimiter
nsgmls:ballads.xml:80:5:E: end tag for "lg" omitted, but OMITTAG NO was specified
nsgmls:ballads.xml:78:9: start tag was here
```

The first of these simply reminds you that you have not specified a DTD, and thereore the parser cannot do more than check the well-formedness of your document. The next four error messages all relate to the same problem, to do with basic XML syntax. (Hint: Remember the rules about ampersands?) The last two point messages confirm that the document is not well-formed: there is a start-tag for the `<lg>` element tag at character position 9 of line 78, but the parser has reached character position 5 of line 80 without finding a corresponding `<lg>` start-tag.

Let's start by sorting out line 61. You can either scroll down the text watching as the line numbers in the middle bar increment, or you can go straight there by typing ESC-g to move the cursor to the minibuffer, then typing 61, and pressing return.

At line 61, you should be able to see what needs fixing: all those instances of `&c.` need to be changed into `&amp;c`. The quickest way of doing this is with the emacs Query Replace command, accessed from the Search menu, or by typing ESC-%. Either form will move you to the minibuffer again, with the prompt `Query Replace:` . Type `&c.` and press Return. The word `with:` appears in the minibuffer. Type `&amp;c` and press Return again. The cursor moves to the first occurrence of the search string in your text, and waits for you to indicate whether or not this should be replaced. Type an exclamation mark to do the replacement throughout the file.

When you think you've got your file well-formed, check it with C-c C-v again! Remember to save the file. To get rid of the window showing output from the parser, just type C-x 1.

3. Creating a new (valid) file with emacs

In this exercise, we'll make a short TEI header for our file, and then build up a complete TEI document that can be properly validated, and which can make use of the XML-specific features of emacs.

We will begin by making a new file, as before. Type C-x C-f, or choose `Open` from the Files menu. When prompted, enter the name ballads*Doc*.xml (or any other name you like, provided that it ends with `.xml`).

From the DTD menu (second from the right) select the second item, `Insert DTD`. A sub menu appears, offering you a choice of predefined DTDs. Choose the option `(XML) TEI Lite`.

Because emacs already knows about this DTD, it can do a lot of the work for you, as you will now see. It will begin by inserting something like the following lines at the top of your file:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="teixlite.css"?>
<!DOCTYPE TEI.2 PUBLIC "-//TEI//DTD TEI Lite XML ver. 1//EN"
"c:/dtds/xml/tei/teixlite.dtd">
```

. These lines you will recognize, probably, as the incantantions necessary to state (a) the document following is an XML document (b) it may be formatted using the teixlite.ccs stylesheet, and (c) it conforms to the TEI Lite XML DTD. Move the cursor down to follow the last of these lines.

You could just start typing in XML at this point. But an easier way is to go to the Markup Menu (fourth from the right) and choose `Insert Element`. As you are at the start of your new document, only one element is legal: `<TEI.2>`. Choose it, and see what emacs does to help you.

Wherever it can, emacs puts in the tags that are required by the DTD automatically. Where the DTD requires something, but gives you a choice, emacs puts in a comment (in red) to show you what the choices are. According to the teixlite dtd you must supply for any document

- a title (that's where the cursor is)
- a publication statement
- a source description
- and, within the text itself, either a `<body>` or a `<group>` element

Type in a title for your new document and then move the cursor to the start of the first red comment. Choose `Insert Element` from the Markup menu again: you will see that the choice of elements available is now different. We suggest you choose `<p>` here. Within the `<p>`, you will find a very large number of elements is available: for simplicity we suggest though that you just type in a piece of text such as `Unpublished exercise`. Then repeat the same procedure for the `<sourceDesc>` element. If you want to explore what options are available with other elements, feel free to do so!

Finally, you can insert the well-formed document you made earlier at the right place in the document (inside the `<text>` element) by putting the cursor there and selecting `Insert file` from the Files menu, or typing C-x i. Either way, you'll be prompted to type the filename in the minibuffer, and press return, as usual.

Now that you have a complete XML document, you should save it and validate it, with C-c C-v. When you do so, you will find that the tag

```
<figure entity="frame01540">
```

is no longer acceptable: this is because you must define the entity frame01540 before referencing it in this way. We've provided a file frame01540.gif containing the image in question: to embed it within your document at the right place, you need to associate the entity name frame01540 with that file, and also to specify that this is a GIF format file. All that is accomplished by the incantation

```
[<!NOTATION GIF SYSTEM ""><!ENTITY frame01540 SYSTEM "frame01540.gif" NDATA GIF>]
```

which needs to be placed *within* the DOCTYPE declaration at the beginning of the file. Take care with those quotes and brackets! The start of your document should now look something like this:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="teixlite.css"?>
<!DOCTYPE TEI.2 PUBLIC "-//TEI//DTD TEI Lite XML ver. 1//EN"
"c:/dtds/xml/tei/teixlite.dtd"
[<NOTATION GIF ENTITY SYSTEM "">
 <!ENTITY frame01540 SYSTEM "frame01540.gif" NDATA GIF>]
>
<TEI.2>
...
```

Just in case you get stuck, we've prepared a valid version of the document for you: it's called ballads-1.xml.

4. Editing a valid document

Satisfying though it is to have a *valid* document, we are still some way from having a *truthful* one! For example, the TEI `<l>` element is meant to mark a single line of verse, not (as here) several of them. If we don't mark the lines of verse properly, we won't be able to process them properly. Fortunately, emacs in xml mode can help simplify this task. Here are some suggestions:

- Use the mouse to put the cursor at the end of a line of verse and select Insert end tag: only one end-tag is possible. (You could achieve the same effect by selecting End current element, or typing c-c /) Then move the cursor to the start of the next line, and select Insert start-tag: choose the right start-tag, and proceed.
- Put the cursor at the end of a line of verse and type `C-c RETURN`. Move to the end of the next line and repeat. Each such step will split the current element, giving you a new verse line.
- Use the search and replace mechanism you used earlier to replace the newlines in the text with the sequence `</l><l>` (but be careful! you don't want to replace *all* the newlines! When asked, instead of typing an exclamation mark, type `Y` or `N` as appropriate)
- Or you could just type the tags in...

As you've noticed, emacs doesn't require you always to insert complete elements (both start and end tags), so it's possible for your document to become invalid. But any emacs command that operates on elements, rather than just tags, won't work unless your document is valid.

When you get tired of splitting verse lines, we suggest you tag a few other things in the poem, such as the proper names. As long as your document is valid, you can use the Tag Region command on the Markup menu to do this. Simply select the name or string you want to tag using the mouse, and then choose Tag Region. A menu of available elements appears.

When you need to insert an attribute value, you can type it in directly, inside the tag, or you can let emacs put it in for you. For example, put the cursor inside the `<l>` start-tag in front of one of those "Heigh Nelly"s. We will need to indicate to the formatter that these lines are to be right-aligned: one simple way is to use the rend attribute to do this. You can edit the tag itself, by typing in the characters "rend="right-align"". Or, more safely, you could choose `Insert Attribute` from the Markup menu: select rend from the submenu, and then enter the string `right-align` in the 'minibuffer' at the bottom of the screen. Press return to add it to the document.

There is a further range of useful XML-specific facilities on the Modify menu: for example

Kill element delete the current element entirely
Untag element remove the tags for the current element
Change element name change the tags for the current element
Fill element fill and indent text content of the current element, removing redundant white space (you can control the amount of indentation using the `File Options` command on the SGML menu).
Edit attributes edit the attributes of the current element

Most of these have keyboard shortcuts, which are given on the menu and also in your crib sheet.

You can use the `Next trouble spot` (C-c C-o) command on the Move menu to move through the document from error to error, fixing them as you go. Check to see when you have fixed all the errors by running C-c C-v again. Remember to save the document (C-x s) before validating it again.

If your document gets into a mess, you can always undo your last action, by choosing `Undo` from the Edit Menu, type C-_ ,or C-x u. If you get irretrievably confused, try C-g to abandon the current editing commands. And, yes, if all else fails, we have prepared a more fully tagged version of the file for you... it's called ballads-2.xml