XML-Editor KXMLEditor

THE X FACTOR

XML is becoming more important all the time and KXMLEditor makes it easier to handle this data format. Frank Wieduwilt shows us how

Freely translated, XML stands for E*X*tensible *M*arkup *L*anguage. It's well suited to the management of structured data, such as smaller databases, and is used as the file format for a huge variety of programs, including word processors (Kword and Abiword), graphics programs (Kontour and Sodipodi) and spreadsheets (Gnumeric and Kspread). Anyone looking at an XML document might be reminded of the source code of Web pages, as the ASCII text in it contains tags in pointed brackets, where there must always be an initial and an end tag. For example:

```
<author>Joe Hacker</author>
```

From the arrangement of the two tags, it becomes clear that the words *Joe Hacker* in the document should be formatted as the name of an author. If we don't want Joe Hacker's name to appear in the text, we could instead use a tag with attributes:

```
<author surname="Hacker" forename="Joe"/>
```

Since in this case no text has to be enclosed by initial and end tags, both are combined by inserting the end symbol before the final bracket, thus */*.

While HTML is concerned with both structuring and formatting a document, XML is concerned exclusively with structure. Formatting is taken care of by a separate template, known as a "Style Sheet". The effects of separating structure and formatting can be seen in the simple database (Listing 1), which we have developed in Figure 1 in an XML-compatible browser (e.g. Konqueror). Without a style sheet it displays the content of the database completely unformatted.

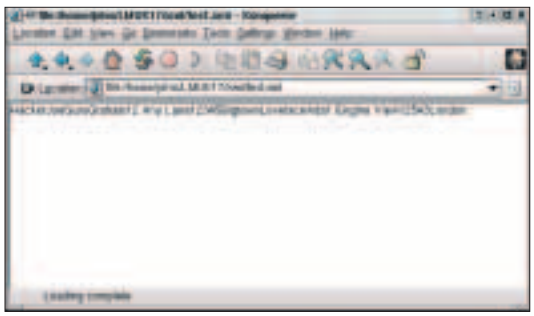

Figure 1: An XML file in Konqueror

XML editors such as KXMLEditor place the file content in a tree-like structure, known as the Document tree, due to the fact that individual elements of an XML file branch out more and more after starting from a root tag. A tag that contains additional sub-tags is referred to as a node.

Style sheets are created using a separate language called XSLT (E*X*tensible *S*tylesheet *L*anguage *T*ransformation). With the help of this you can transform the content of an XML document into various output formats, such as HTML or **PDF**. Sometimes an XML file also needs a DTD, or *D*ocument *T*ype *D*efinition, in which the use of the individual tags are defined. For example, with the aid of a DTD it's possible to check whether an XML document is only using attributes that are permitted for a specified tag, or whether tags are being interlocked which aren't allowed to be interlocked with each other. A good introduction to working with XML can be found at *http://www.webdeveloper.com*

As with any ASCII mark-up format, XML can be written with any word processing program of your choice. KXMLEditor makes it easier to create and to edit XML documents, as its tree representation offers the user an overview of the document tree at all times. It's very easy to copy entire branches within the document. However, KXMLEditor does not yet create DTDs or style sheets.

PDF: *P*ortable *D*ocument *F*ormat. A file format for exchanging formatted text.

Installation

To compile off your own bat, you will need the file kxmleditor-0.7.1.tar.gz from the program homepage or our coverdisc. Unpack this file with the command:

```
tar -xzvf kxmleditor-0.7.1.tar.gz
```

and change to the newly created directory *kxmleditor-0.7.1*. Install the program there by entering *./configure*; start compilation with *make* and finally install it as *root* by entering *make install*.

A 750MHz AMD Duron using Red Hat 7.1 was used as the test system for this article.

XML assistant

The KXMLEditor program can be found at *http://kxmleditor.sourceforge.net*. Red Hat 7.x users

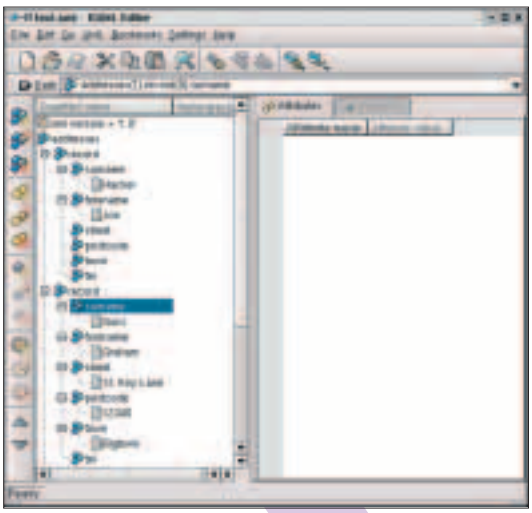

Figure 2: KXMLEditor

can play in the appropriate rpm packet as root with the command

```
rpm -Uvh kxmleditor-0.7.1-0.1.RH7.i686.rpm
```

The installation boxout explains how to install the software from the **source text**. To start the program, enter *kxmleditor &* in a console or select *Applications/KXMLEditor* in the *K* menu.

The KXMLEditor program window is split into three sections (Figure 2). Under the menu line there are two toolbars; the upper one is the general toolbar whilst the one underneath is a navigation line, which displays the active element in the XML file. In the lower part of the program window you'll see the XML toolbar on the left; a tree view in the middle, in which the content of the opened file is displayed; and on the right is an input area with two tabs for the parameters of the tags and their contents.

The interface elements are currently only marked in English, but work is in progress on translation into other languages. Figure 3 explains the meaning of each of the buttons on the XML toolbar.

Editing XML files

The simplest way to familiarise yourself with KXMLEditor is to practise working on an existing XML file. To do this, select *File/Open* from the KXMLEditor menu and load a file from Abiword, Kword or a similar XML-based application. Its structure will now be displayed in the tree view. In Figure 4 the tag pagesize is selected. In the right-hand pane of the work area, under the Attributes tab, you'll see which attributes this tag possesses and their values.

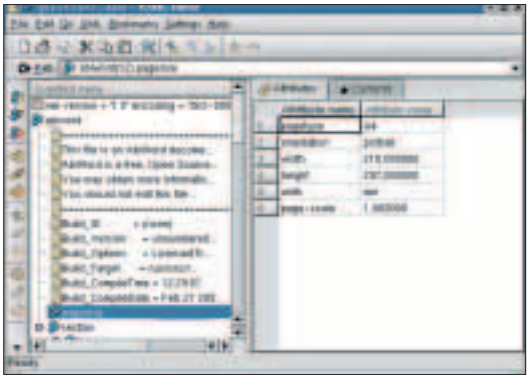

Figure 4: An AbiWord file in KXMLEditor

As an example of a new XML file, we'll use an address database, as address details have a structure in which the individual attributes of forename, surname, street, postcode, town and telephone number are assigned to respective persons. To do this, first create a new XML document with *File/New*.

Every XML file must start with the line:

```
<?xml version="1.0"?>
```

which is known as the prologue. To enter this, select *XML/Insert proc. instruction* from the menu, or press Ctrl+R. In the dialogue that appears, enter *xml* in the Target field and *version = "1.0"* in the Data field. Click OK and the first line of your document will be created in the tree view.

After the prologue, insert the root element of your XML file. Each XML file can have only one such root element, from which all additional elements are derived. In our example the root element is called addresses.

Select *XML/Insert Element* from the menu or press Ctrl+E. In the dialogue which now appears (see Figure 5), enter the name of the element in the Element field. In the Insert: list at bottom left, select root, in order to mark the element as root element.

After this preliminary work, the elements for the records can be defined. To do this, select the addresses element with the mouse, and insert an additional element named "record". To insert an element at the top of the document, click on the Insert drop-down list in the XML Element dialogue, and select "at top". You can define all of the elements in your individual data fields in this manner until you see a complete record in the tree view, as in Figure 6.

You can now copy this empty record by marking the record tag and selecting *Edit/Copy* from the menu, or by pressing Ctrl+C. Select the database tag

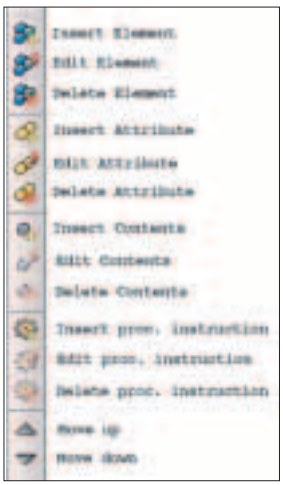

Figure 3: The functions of the XML toolbar

source text: The files containing the program commands in a programming language. In order for the program to be executed by the computer, this must first be translated (or "compiled").

Well-formed and valid XML

The terms "well-formed" and "valid" crop up constantly in connection with XML. An XML document is well-formed when it has no DTD of its own, but each tag is completed by an end tag. However, it is not valid until it has a DTD, whose rules its content obeys.

With KXMLEditor you can only create well-formed XML documents, since it does not offer the option of programming a DTD or of linking an XML file to an already existing DTD.

Info

Information on XML
http://www.webdeveloper.com/xml

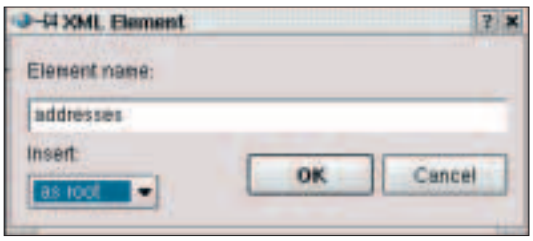

Figure 5: Inserting an XML element

and insert a new, identically constructed record with the command *Edit/Paste*, or by pressing Ctrl+V. The database now contains two records, which are both still empty.

To fill the data fields with content, mark the desired field and select *XML/Edit content* from the menu. In the dialogue box, enter your information in the Content field – this information can cover more than one line. From the Type drop-down list, select whether the data entered is to be considered as text, a comment or CDATA (Character Data). Before displaying your new addition, the XML parser checks to see whether the content and defined type correspond with each other; comments are ignored when the file is outputted. If you select the CDATA type, the parser checks to see whether there really are only numbers and letters present. If text is chosen this check is left out.

You can now gradually fill all the data fields with content and insert new records as required. The sequence of the records can be changed by selecting a record tag and altering its position with the aid of the two buttons Move element up and Move element down, from the XML toolbar.

If you've followed the examples so far, you should now have a well-formed XML file (Listing 1),

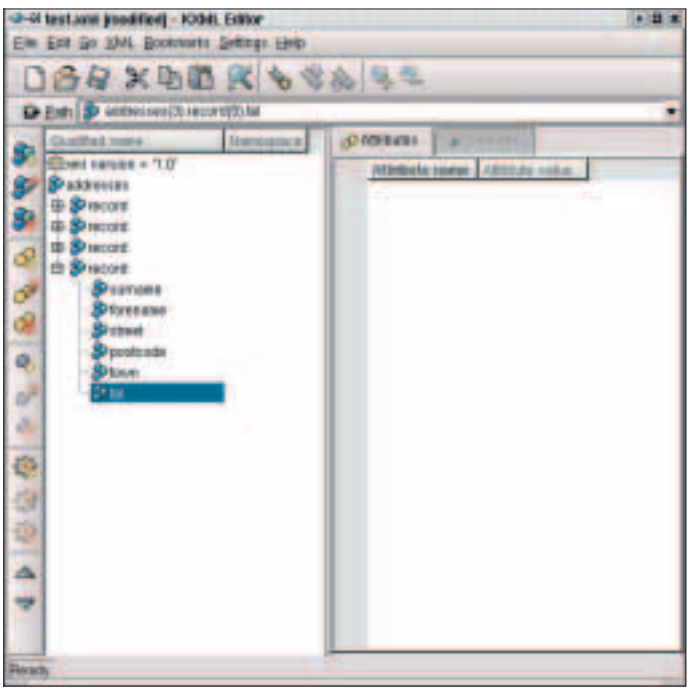

Figure 6: A complete record

Table 1: Important key combinations

Action	Key shortcut
Open file	Ctrl+O
Save file	Ctrl+S
Close file	Ctrl+W
End program	Ctrl+Q
Insert element	Ctrl+E
Insert attribute	Ctrl+A
Insert content	Ctrl+C
Set bookmark	Ctrl+B

although you won't have a DTD or style sheet as yet. You must create both of these in a text editor of your choice and insert them in the XML file by hand, as KXMLEditor does not yet have a corresponding functionality.

Strengths and limitations

The program is therefore suitable for viewing and editing small XML files, but not for large databases or heavily formatted documents. KXMLEditor is very stable and is easy to use via the keyboard. In the program help, all program functions are explained; but there is no introduction to XML, which is necessary to understand the software.

Listing 1: The example database

```
<?xml version = '1.0' ?>
<addresses>
   <record>
        <surname>Hacker</surname>
        <forename>Joe</forename>
        <street></street>
        <postcode></postcode>
        <town></town>
        <tel></tel>
   </record>
   <record>
        <surname>Guru</surname>
        <forename>Graham</forename>
        <street>12 Any Lane</street>
        <postcode>12345</postcode>
        <town>Bigtown</town>
        <tel></tel>
   </record>
   <record>
        <surname>Lovelace</surname>
        <forename>Ada</forename>
        <street>1 Engine View</street>
        <postcode>12543</postcode>
        <town>London</town>
        <tel></tel>
   </record>
</addresses>
```