XUL and the building processe of a UI

Andre Deienno Pansani,
superviser: prof. Peter Forbrig,
Departament of Computer Science,
University of Rostock, Germany

June 13, 2003

Abstract

With the right resources it is possible to build a User Interface(UI) in an efficient way. The XML User Interface Language (XUL) could be useful for designing dialog or presentation models. The aim of this document is to present UI editors and how to transform from the XML of the editor to a XUL UI.

1 Introduction

Two User Interface editors were chosen: Glade and QtDesign. The Glade is a UI builder for GTK+ released under the GNU General Public License (GPL). The QtDesign editor provides many functionalities needed to build state-of-the-art graphical user interfaces. The initial ideas of using one of the both editors and then convert to a XUL description comes from the LUXOR XUL project [1].

Visualising XUL with Mozilla browser

The Mozilla [2] browser supports XUL visualisation. In reality, the whole Mozilla is also based on XUL. In order to visualise any XUL file in the browser it is necessary to add the following namespace, *xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"*.

See the example below:

```
<?xml version="1.0" encoding="utf−8"?>
<xul
xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
...
</xul>
```

2 The Glade editor

The GLADE editor [3] designs the UI with its on XML and saves it in a glade file(.glade). The editor is not just defining an abstract UI but it can also generate code. The code is written in C-language and can later be compiled in an executable file with another tool called GTK+.

GTK+ is a multi-platform toolkit for creating graphical user interfaces, offering a complete set of widgets. GNU Lesser Public License (LGPL). It can be found at *http://www.gtk.org*.

Tansformation to XUL

The main idea is to use Glade to build forms, menus, toolbars, other UI stuff drag-and-drop style and then transform the GLADE XML to XUL. Glade-To-XUL is an experimental XSL/T stylesheet that transforms Glade XML UI descriptions to XUL XML UI descriptions. The current version does not support all widgets. It can be found in the LUXOR XUL Home Page [1] at "Download Contrib".

To use Glade-To-XUL pass in your Glade XML UI description to an XSL/T engine such as Apache Xalan using the glade2xul.xsl stylesheet.

3 QtDesign editor

3.1 General information

The QtDesign is a powerefull GUI builder because it has a preview mode that allows you to see your dialog 'for real' right away, without any compiling. The specific name of the tool used in this work is "Qt/Windows Non-commercial (version 2.3)". It is a product of TROLLTECH [4] under the terms of GPL and it comes from the "Qt C++ GUI Application Development Toolki family". The following versions are currently avaliable:

- Qt 3.2 Beta 1
- Qt 3.1
- Qt 3.0
- Qt 2.3 (the only non-commercial version)

There are differences in XML User Interfece description between the disponible versions. So the software developed may work only with the correspondent editor(version 2.3). But a XSL tool is on development that make transformations between the different versions of Qt User Interface descriptions. This tool its a part of the "User Interface Compiler" project [5].

3.2 The first step was ready

Again, from the LUXOR project came the first step to convert a XML UI description from an avaliable GUI Builder in XUL. A tool called "qatar-1.0-b1" can be found at "Download Contrib" of the LUXOR XUL Home Page [1]. It supports the following widgets:

- Form (window)
- Group Box (groupbox)
- Layout Out Horizontally (hbox)
- Layout Out Vertically (vbox)
- Check Box (checkbox)
- Line Edit / Multi Line Edit (textbox)
- Push Button (button)
- Text Label (label)
- Line (separator)

3.3 The Rostock University contribution

In order to fulfil a User Interface, the supported widgets from "qatar-1.0-b1" were not enough. Therefore the following items were implemented, taking the XUL standard as a model at Xul Planet Home Page [6]:

Tabwidget (tabbox)

Due to the necessity of a Tabbox this widget was implemented.

Tabbox Model:

```
<tabbox id="tablist">
<tabs>
– tab elements go here –
</tabs>
<tabpanels>
– tabpanel elements go here –
</tabpanels>
</tabbox>
```

List View (tree)

This widget involved a research of what method would be the best to keep the tree elements. The project decision was to keep all the information of the tree in a RDF format. As the result, a RDF file was created with tree information and only a template is included on the output XUL file.

Tree Model:

```
<tree id="my-tree" flex="1" datasources="rdf:files" ref="file:///" flags="dont-build-content">
<treecols>
<treecol id="Name" label="Name" primary="true" flex="1"/>
<splitter/>
<treecol id="Date" label="Date" flex="1"/>
</treecols>

<template>
<rule>
<treechildren flex="1">
<treeitem uri="rdf:*">
<treerow>
<treecell label="rdf:http://home.netscape.com/NC-rdf#Name"/>
<treecell label="rdf:http://home.netscape.com/WEB-rdf#LastModifiedDate"/>
</treerow>
</treeitem>
</treechildren>
</rule>
</template>

</tree>
```

Pixmap Label (image)

Unfortunately the editor saves the image in the UI file, instead it just keeps its path name. The solution found, in order to use this feacture, was: add the image path manually at the *"src"* attribute in the *"image"* element direct in the XUL file.

The *"box"* element contains the *"image"* because of alignment reasons. It should keep the size of the image constant instead let it be resized by the *"flex"* attribute of a "container element"(*hbox*, *window*, ...). It is possible to set the size of the image using the QtDesign. The *"scaledContents"* property of the image is the resize attribute. When it is "true", the image at the XUL file will have exactly the size (in pixels) of the image at the editor. When it is "false", the image will have its own size, at the XUL file.

Image Model:

```
<box align="center">
<image src="image_path"/>
</box>
```

Text Browser (iframe)

This is a powerful widget which makes possible to visualise different files. It can be used to visualize HTML, in this way the UI description has its own browser. But actually, it can be used to visualize any kind of file supported by the browser, in our case the Mozilla [2] browser.

Browser Model:

```
<iframe id="iframe_id" src="source_path" flex="1"/>
```

4 About Flexibility

Due to the absence of positions information, the manually corrections of the Layout are essencial. As each User Interface has its own specialities, the automatic code generation becomes insufficient. Therefore it is important to learn about flexibility.

XUL lays out elements on a window by calculating suitable widths and heights for the elements and then adding space where they are flexible. Unless you specify information about the width and height of an element, the default size of an element is determined by its contents. If you create a button with a very long label, the button's default size will be large enough to hold the entire label. Other elements, such as the text box have chosen a suitable default size.

The *flex* attribute is used to specify if an element can change size to fit at its "container element". The *flex* attribute can be applied to any element.

What if you want to set one element to grow twice as large an another? You can use a higher number as the value of the *flex* attribute. The values of the *flex* element are a ratio. If one element has a *flex* of 1 and the next one has a *flex* of 2, the second one will grow at twice the rate of the first one. In effect, a *flex* of 2 says that this element has a *flex* that is two times the elements that have a *flex* of one.

The *flex* attribute only hints as to what the size ratio between elements should be. For various reasons, including explicit sizes set on elements, or special requirements of certain widgets, the flexibility may not be honored exactly. More information at Xul Planet [6].

5 Conclusion

It is more than clear that UI can be well described with the help of XML. Many tools are already avaliable to work with the building processes of a UI, and XUL(XML User Interface Language) it is a good alternative.

References

[1] Luxor - XML User Interface Language (XUL) Toolkit, *http://luxor-xul.sourceforge.net*

[2] Mozilla, *http://www.mozilla.org/*

[3] GLADE (GTK+ User interface Builder), *http://glade.gnome.org*

[4] TROLLTECH Home Page, *http://www.trolltech.com*

[5] UICompiler, *http://uic.sourceforge.net*

[6] Xul Planet, turorial with examples, *http://www.xulplanet.com/*