Distributed Software Development

XUL

Chris Brooks

Department of Computer Science
University of San Francisco
XUL is an XML-based language designed for building user interfaces within Mozilla (and Firefox).
  - Pronounced “zool”

Uses Javascript for event handling, CSS for display

Provides a rich set of graphical widgets

Can either be loaded as a URL, or installed.
  - Firefox’s extensions are written with XUL.
The primary XUL element is a window
- Similar to a frame in Java
- Elements are placed within the window.
14-2: Example

<?xml version="1.0"?>
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<window
  id="window1"
  title="A sample window"
  orient="horizontal"
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
</window>
Elements (widgets) are then placed within the window.

XUL has all of the typical elements you’d expect, plus some others.

Each element is indicated by an XML element.

Images can also be included as elements.
<window
    id="findfile-window"
    title="Find Files"
    orient="horizontal"
    align="start"
    xmlns:html="http://www.w3.org/1999/xhtml"
    xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
    <button
        id="identifier" class="dialog"
        label="OK" default="true" />
    <button label="Not OK" />
    <button label="cancel"/>
    <label id="label1" value="here is a label"/>
</window>
14-5: Other elements

- Lists
- Trees
- menus
- spacers
- Status bars
- toolbars
- and many, many more
Since XUL is an XML-based language, we can use CSS to control the visual display of elements.

- Keeps the XUL simple(r), separates presentation from function
14-7: Using CSS to control display

XUL code:

```xml
<image id="image1"/>
<image id="search"/>
```

Style Sheet:

```css
#image1
    list-style-image: url("chrome://findfile/skin/banner.jpg");

#search
    list-style-image: url("chrome://findfile/skin/images/search.jpg");
```
XUL uses what’s called the ’box model’ to lay out elements.

- This is somewhat like Java’s Flow Layout Manager

- Objects within a box are arranged horizontally or vertically.

- Boxes can be nested.
<hbox>
<button id = "hb1" label="Horiz button 1" />
<button id = "hb2" label="Horiz button 2" />
</hbox>

<vbox>
<button id = "vb1" label="Vert button 1" />
<button id = "vb1" label="Vert button 1" />
</vbox>
You can also use CSS to specify element sizes.

Boxes and elements can also be set to be flexible

```html
<hbox flex="1">
  <button label="Yes" flex="1"/>
  <button label="No"/>
  <button label="I really don’t know one way or the other"/>
</hbox>
```
You can include HTML formatting inside XUL

HTML requires its own namespace for HTML tags

```xml
<window
    id="findfile-window"
    xmlns:html="http://www.w3.org/1999/xhtml"
    xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
    <html:ul>
        <html:li> blah </html:li>
        <html:li> blah </html:li>
        <html:li> blah again </html:li>
    </html:ul>
</window>
```
14-12: Event Handling

- Normally, you will specify Javascript functions to execute when an event is received by an element.
- Typically, your XUL file will reference a separate file containing Javascript.

```xml
<window
   id="findfile-window"
   xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
  <script src="test1.js" />
  <button
     id="identifier" label="OK"
     oncommand="doStuff();" />
</window>
```

```javascript
## test1.js
function doStuff()
var l = document.getElementById('label1');
l.setAttribute("value", "here's some content");
```
Notice that in the previous Javascript, we accessed an element by finding its ID.

Since XUL is XML, we have access to the DOM tree.

This makes it easy to retrieve elements, just as we’ve done previously.

You can also use this to alter the UI on the fly by changing the DOM tree.