X-Forms in X-Smiles

Mikko Honkala

Helsinki University Of Technology, Finland

Mikko.Honkala@hut.fi

http://www.x-smiles.org/
Outline of the Presentation

• Introduction
  – HIIT XML Devices project
  – W3C’s XForms
• XForms working draft
• XForms implementation in X-Smiles browser
  – Demos
HIIT XML Devices

• Researching the use of XML in small embedded devices
  – Handhelds, Digi-TV, NG mobile phones
• Porting X-Smiles XML browser into different devices
• Enhancing client-side interactivity through W3C’s XForms
Introduction to XForms

- World Wide Web Consortium (W3C) creates Web standards.
  - HTML, XML, SVG, SMIL, etc...
  - over 500 member institutions.
  - co-steered by MIT (USA), KEIO (Japan) and INRIA (France). (vendor and market neutral).

- XForms.
  - W3C’s future Web forms technology.
  - Ideas from proprietary form languages (FML, Formsheets, XFA, XFDL).
  - Builds upon tested, pre-existing XML technologies rather than re-inventing the wheel.
  - X-Smiles in XForms WG
### Purchase Order

<table>
<thead>
<tr>
<th>Units</th>
<th>Item</th>
<th>Price/unit</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>X-Smiles desktop</td>
<td>50 mk</td>
<td>150 mk</td>
</tr>
<tr>
<td>1</td>
<td>X-Smiles PDA</td>
<td>500 mk</td>
<td>500 mk</td>
</tr>
<tr>
<td>1</td>
<td>Java debugger</td>
<td>1500 mk</td>
<td>1500 mk</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtotal</td>
<td>2150</td>
</tr>
<tr>
<td>Taxes</td>
<td>473</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2623</strong></td>
</tr>
</tbody>
</table>

Submit
What’s missing in the current Web forms

- No separation between content and presentation
- Validation must be done at the server: more round-trips
- Constraints and calculations between fields have to be programmed with scripts
- Most web services live in the XML world. Therefore a component is needed to map HTML forms into XML and vice versa
How does XForms solve these problems?

• Content is separated from the presentation
• Presentation is defined elsewhere in the document, it is only bound to the content
• Validation is done in the client using XML schema and inter-data constraints
• Constraints and calculations are defined declaratively in the markup. Let the XForms processor implement them rather than program them in JavaScript
• XForms capable client receives and sends XML directly
**XForms document components**

- **Instance** – content (XML instance)
- **Model** – validation, constraints, calculations
- **User Interface** – embedded in host language
- In addition:
  - **Binding** – binding between the instance, model and UI
- Can be seen as a MVC
XForms Use Cases

- Multipage Tax Input forms with calculations and validation
- Web shop order forms
- Interactive Vector Graphics
- Software configuration frontend
  -> ... everything that needs user interaction within Web document formats
## Purchase Order

<table>
<thead>
<tr>
<th>Units</th>
<th>Item</th>
<th>Price/unit</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>X-Smiles desktop</td>
<td>50 mk</td>
<td>150 mk</td>
</tr>
<tr>
<td>1</td>
<td>X-Smiles PDA</td>
<td>500 mk</td>
<td>500 mk</td>
</tr>
<tr>
<td>1</td>
<td>Java debugger</td>
<td>1500 mk</td>
<td>1500 mk</td>
</tr>
</tbody>
</table>

**Subtotal**: 2150

**Taxes**: 473

**Total**: 2623

Submit
XForms Example: Instance Data

```xml
<purchaseOrder>
  <items>
    <item>
      <name>X-Smiles desktop</name>
      <units>2</units>
      <price>50</price>
      <total>0</total>
    </item>
    <item>
      <name>X-Smiles PDA</name>
      <units>2</units>
      <price>100</price>
      <total>0</total>
    </item>
    ...
  </items>
  <totals>
    <subtotal>0</subtotal>
    <tax>0</tax>
    <total>0</total>
  </totals>
  <info>
    <tax>0.22</tax>
  </info>
</purchaseOrder>
```
XForms Example: Model

```xml
<head>
  <xfm:xform>
    <xfm:model href="purchaseOrder.xsd" />
    <xfm:instance href="purchaseOrderData.xml" />
    <xfm:bindings>
      <xfm:bind ref="purchaseOrder/totals/subtotal"
        calculate="sum(../..//items/item/total)"/>
      <xfm:bind ref="purchaseOrder/totals/tax"
        calculate="../subtotal * ../..//info/tax"/>
      <xfm:bind ref="purchaseOrder/totals/total"
        calculate="../subtotal + ../tax"/>
      <xfm:bind ref="purchaseOrder/items/item/total"
        calculate="../units * ../price"/>
    </xfm:bindings>
  </xfm:xform>
</head>
```
XForms Example: The user interface

```xml
<body>
  <table>
    <th><td>Units</td><td>Item name</td><td>Price</td><td>Total</td></th>

    <xfm:repeat select="/purchaseOrder/items/item">
      <tr>
        <td>
          <xfm:textbox ref="units">
            <xfm:hint>Enter the quantity of this item.</xfm:hint>
          </xfm:textbox>
        </td>
        <td><xfm:output ref="name" /></td>
        <td><xfm:output ref="price" /></td>
        <td><xfm:output ref="total" /></td>
      </tr>
    </xfm:repeat>
  </table>

  <xfm:output ref="/purchaseOrder/totals/total">
    <xfm:caption>Total price</xfm:caption>
  </xfm:output>
</body>
```
Validation Using XML Schema

• XML Schema is a W3C recommendation from Summer 2001
  – Defines the structure of an XML document as well as datatypes
    • ’xsd:date’ (1999-05-31)
    • ’xsd:time’ (13:20:00.000)
    • ’xsd:decimal’ (-123.4)
  – Datatypes can be created by the user with restrictions and unions
    • e.g. An integer smaller than 1000
  – Datatypes are more important to Xforms
    • It is also possible to use a simpler ’schema for instance’ syntax
Constraints

- Constraints apply to instance data in the model
  - calculate – parts of data calculated from other parts
  - relevant – is the item shown to the user
  - readOnly – can the user edit the item
  - isValid – is the data valid
  - required – is the data required for submission

```xml
<bind ref="items/item/total"
     calculate="../units * ../price"
     relevant="../units>0" />
```
Binding & Constraints using XPath

- XPath is a W3C recommendation
  - Developed mainly for XSLT but very general
- is used to:
  - select a single node from an XML document
    `/purchaseOrder/items/item[1]`
  - select multiple nodes (nodeset) from an XML document
    `/purchaseOrder/items/item`
  - perform calculations with the data in XML
    `sum(/purchaseOrder/items/item/total)`
XForms User Interface

- Very general level of user interface controls
  - selectOne
  - selectMany
- Can be used also in non-graphical environments (e.g. VoiceXML)
- Meant to be embedded in XML languages
- Additional presentation hints with CSS stylesheets and presentation parameters

```xml
<selectOne selectUI="listbox" style="width: 200pt; height: 100pt; background-color: gray;"/>
```
X-Smiles Browser

- XML Browser
  - XSL FO, SVG, SMIL, Xforms.
  - XSLT Transformations
  - ECMAScript
- Java-based
  - portability, available components, JMF
- Open Source
- Virtual prototype
  - Desktop, digi-TV, PDA, mobile phone
XForms Implementation in X-Smiles

- The first browser implementation
- Supports most of the XForms features:
  - user interface controls (textbox, selectOne, ...)
  - validation
  - calculations
  - mouse events
- We are co-specifying XForms
Embedding in different Markup languages

- It is possible to embed XForms in all XML languages supported by X-Smiles
  - SVG
    - inside ’foreignObject’ element
  - SMIL
    - As a content object
  - XSL FO
    - Embedding within ’fo:declarations’ and ’fo:instream-foreign-object’
Implementation details

- Schema validation is done using Xerces (Apache’s XML parser)
- XPath calculations use Xalan’s Xpath engine (Apache’s XSLT transformer)
- User interface implemented using Swing widgets
- Calculation order and circularity checking algorithms implemented natively
# XForms Features in X-Smiles

<table>
<thead>
<tr>
<th>Feature</th>
<th>Now</th>
<th>Future</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form controls</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Datatype aware controls</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Schema validation</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Inline/Ext instance &amp; schema</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Host languages</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Constraints: calculate</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>readOnly</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>relevant</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>required</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>validate</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Switch</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Repeat</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>
Future work

- Implement XForms processor on top of DOM
- Constraints: readOnly, relevant, required, validate
- Using datatype information in form controls
  - This information not currently available from Xerces
- Repeat
- Switch
- uploadMedia – submitting binary within XML data
- Styling with CSS
Demos

- X-Smiles configuration file
- XForms in SMIL
- XForms in SVG / Scripting
- XForms calculations