XML basics

XML stands for Extensible Markup Language. It is a computer language very similar to HTML. It works by way of “tags”. There are some ‘self closing’ or single tags but most tags ‘open’, surround something and ‘close’. Tags can have attributes as part of the opening tag. In mResearch many of the tags are our own - designed specifically for our software.

Tags appear in angle brackets like "<" and ">". Here is an example of an HTML tag for making the word “Flower” appears in bold:

    <b>Flower</b>

The closing tag contains a forward slash.
Here is the XML in mResearch to format a single item horizontally so that it appears similar to the table style of a “matrix” item.

    <item itemRef="7">
        <settings>
            <renderAsBlock>1</renderAsBlock>
        </settings>
    </item>

There are three tags in this sample; the ‘item’ tag (which has “itemref” as an attribute), the ‘settings’ tag and the ‘renderAsBlock’ tag. In order for this code to work for an m/survey in mResearch it must be in two more tags. All XML code must between the open and close of the ‘Survey’ tag. After that tags must be in one of the categorical tags; ‘settings’, ‘items’, ‘filtering’ or ‘branching’. You can put as many tags in the categorical tags as you want but they appear only once. So to use the above code you’d have to put this in mResearch:

    <survey>
        <items>
            <item itemRef="7">
                <settings>
                    <renderAsBlock>1</renderAsBlock>
                </settings>
            </item>
        </items>
    </survey>

The ‘settings’ tag above is not the categorical tag ‘settings’ (which is for global survey settings) but is within the ‘item’ tag and relates to settings for the item referenced in that tag. In order to reduce confusion, tags in HTML and XML are indented to represent the level of the tag (each tag goes back one further tab). This convention does not affect the operation of the tags but is for readability and organization.
Chapter 2

Thinking like a programmer

Using XML in an m/survey is different from using some of our more user friendly interfaces like the survey designer of mResearch. Using XML will require that you think a bit more like a programmer. In order to help you think a little more like a programmer let’s briefly discuss syntax, flexibility and quality control.

Syntax:
Computer languages are very unforgiving. Even a slight deviation, like an extra comma, can “break” the code. The exact “syntax” must be followed to get the desired outcome. If you write a lot of XML and your survey shows an error it may be difficult to identify the cause. The usual culprit will be syntax.

Flexibility:
Usually when you use a program all the choices you can make have been tested and anything that wouldn’t work or that hadn’t been thoroughly tested wouldn’t be available to you. Programming is different. You can easily find yourself using the XML for things we never thought of or in combinations we haven’t tested. This code has been thoroughly tested for what the sample survey is doing but writing your own code includes the ability to pave new roads. We will of course help to navigate any new territory.

Quality control:
QC is a part of every project but the kind of QC that is needed when you have the type of flexibility described above is different. The difference is that you have to think of ALL of the possible effects of your code including unintended effects to seemingly unrelated things. If, for instance, you use our new branching logic every possible branch should be checked including what happens if you skip the question and whether the skipped questions are part of the logic needed for anything else and so on. We will help you to design the proper QC for your projects that include XML.
Chapter 3

What follows is the XML code that makes the sample survey run. The headers in red do not belong but are there for description only. They don't belong in the code. The headers exactly match the red section headers in the sample survey.

<survey>
  
  <settings>

  <Clear Form Button>
  <showClearFormLink>true</showClearFormLink>

  <Required Indicator>
  If a question is required, you can add special text identifying the requirement
  <requiredIndicator>*Response Required</requiredIndicator>

  <Progress Meter>
  Only two progress meters are supported in a survey header.
  <progressMeter>
    <showItemText>true</showItemText>
    <showItemPercent>true</showItemPercent>
    <showItemGraph>true</showItemGraph>
    <showPageText>true</showPageText>
    <showPagePercent>true</showPagePercent>
    <showPageGraph>true</showPageGraph>
  </progressMeter>
  </settings>

  <items>

  <Text box validated against a list>
  <item itemRef="1">
    <itemType>4</itemType>
  </item>

  <Series of textboxes for ranking>
  <item itemRef="2">
    <itemType>5</itemType>
    <settings>
      <minValue>1</minValue>
      <maxValue>3</maxValue>
      <allowRepeats>0</allowRepeats>
      <requireEachValue>0</requireEachValue>
    </settings>
  </item>
  </items>
Render as block
You can also add in a required indicator into the block using `<requiredIndicator>`.

```
<Item itemRef="7">
  <Settings>
    <RenderAsBlock>1</RenderAsBlock>
    <RequiredIndicator>*Required</RequiredIndicator>
  </Settings>
</Item>
```

Series of text boxes that must equal a certain amount

```
<Item itemRef="8">
  <ItemType>6</ItemType>
  <Settings>
    <Total>100</Total>
    <Tolerance>0</Tolerance>
  </Settings>
</Item>
```

Dual scale survey
Note: Dual surveys do not support checkbox items.

```
<Item itemRef="12">
  <ItemType>7</ItemType>
  <Item itemRef="15"/>
</Item>
```

The next four questions are in random order.

```
<Item itemRef="18">
  <Settings>
    <RandomizeType>1</RandomizeType>
  </Settings>
</Item>

<Item itemRef="19">
  <Settings>
    <RandomizeType>1</RandomizeType>
  </Settings>
</Item>

<Item itemRef="20">
  <Settings>
    <RandomizeType>1</RandomizeType>
  </Settings>
</Item>

<Item itemRef="21">
  <Settings>
    <RandomizeType>1</RandomizeType>
  </Settings>
</Item>
```
Filtering: Make an item disappear

<filtering>

<item itemRef="23">
  <condition>
    <item itemRef="22" inverse="true">
      <response value="2" />
      <response value="3" />
    </item>
  </condition>
</item>

<item itemRef="24">
  <condition>
    <item itemRef="22" inverse="true">
      <response value="1" />
      <response value="3" />
    </item>
  </condition>
</item>

<item itemRef="25">
  <condition>
    <item itemRef="22" inverse="true">
      <response value="1" />
      <response value="2" />
    </item>
  </condition>
</item>

</filtering>

Branching - Advanced features

<branching>

<item type="basic" itemRef="26">
  <response value="none" destinationItemRef="end" />
  <response value="1" destinationItemRef="end" />
  <response value="3" destinationItemRef="28" />
</item>

</branching>

</survey>
Preselect default value

<survey>
  <items>
    <item itemRef="1">
      <settings>
        <defaultQValue>3</defaultQValue>
      </settings>
    </item>
    <item itemRef="2">
      <settings>
        <defaultWValue>ABC</defaultWValue>
      </settings>
    </item>
    <item itemRef="2">
      <settings>
        <defaultQMValue>1,2,4</defaultQMValue>
      </settings>
    </item>
  </items>
</survey>
Chapter 4
Here is the path to the m/survey companion pages and a sample companion page.

1. First change your survey to one of the "m/survey" templates in the survey designer of mResearch:

   ![Survey Designer Screenshot]

   - Select the "m/survey" template in the survey designer.

2. Starting from the main menu navigate to the companion page for the survey (Main Menu→Utilities and Advanced Modules→m/survey Custom Features→then select your survey from the drop down):

   ![Main Menu Screenshot]

   - Navigate to the companion page for your survey.

© 2011-2013 Modern Survey, Inc. All Rights Reserved
### Utilities and Advanced Modules

**Set Custom Mapping**  
Map survey questions to custom values within distribution

**Run Custom Mapping**  
Update survey scores with mapped custom values

**Export to Excel**  
Export group raw data to Excel

**Pulse Survey / Meta Events**  
Group several events together into a single Meta Event

**Import Syntax Groups**  
Import Syntax Groups from Microsoft Excel

**mSurvey Custom Features**  
Modify Survey Header, Page Title, Language, Finish Text, and CSS

**Alerts**  
Manage Alerts and Alert Contacts

---

### Modify Survey Data

<table>
<thead>
<tr>
<th>Select Survey</th>
<th>Survey Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select the survey to modify.</td>
<td>Select a survey template for this survey.</td>
</tr>
<tr>
<td>- Select Survey -</td>
<td>- Select Survey Template -</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Master Survey</th>
<th>Survey Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select the survey that this survey is a translation of, if applicable.</td>
<td>Select the language to use for the button text on the survey.</td>
</tr>
<tr>
<td>- Select Master Survey -</td>
<td>- Select Language -</td>
</tr>
</tbody>
</table>

**Page Title Text**  
This text will appear in the browser’s title bar for all pages of the survey.

**Survey Name**  
Select a new name for the survey.

**Survey Theme**  
Select the look of your survey. Additional styling can be added using Survey CSS (below).

- Select Survey Theme -
Chapter 5
This is a quick guide to how changes you can make to the sample code would change your survey. With this information you should be able to repurpose this code for other projects.

Clear Form Button

If this code is included, the clear form button will appear. There are no adjustments.

Progress Meter

"true" for any of these options turns the option on. Only two progress meters are supported in the header of a survey.

```xml
showItemText = "Item 8 of 23"
showItemPercent = "14% Complete"
showItemGraph = "14%"
showPageText = "Page 2 of 7"
showPagePercent = "30% Complete"
showPageGraph = "30%"
```

Text box validated against a list

1. Set up a drop down list in the survey designer of mResearch
2. The survey taker will see a textbox and must type in something identified as a response option for the drop down item.
3. Identify the item as a type “4” in XML on the companion page:

```xml
<Item itemRef="1">
  <itemType>4</itemType>
</item>
```

Series of textboxes for ranking

```xml
<Item itemRef="2">
  <itemType>5</itemType>
  <settings>
    <minValue>1</minValue>
    <maxValue>3</maxValue>
    <allowRepeats>0</allowRepeats>
    <requireEachValue>0</requireEachValue>
  </settings>
</item>
```

1. (Survey designer) Make a new radio item for each option to be ranked
2. (Survey designer) Ignoring scale use the item text for the option
3. (Survey designer) Use a section header for the item text
4. (Survey designer) Put a section break (or page break) after the last item to be ranked
5. (XML on the companion page) Identify the Q number of the first item
6. (XML on the companion page) Set the minValue and maxValue of the possible ranks – (1st, 2nd, 3rd in the above)
7. (XML on the companion page) “AllowRepeats” is either a ‘0’ for don’t allow or a ‘1’ for allow (if allowed you could have more than one item marked as 1st for instance)
8. (XML on the companion page) “requireEachValue” is either a ‘0’ for when you need each rank to be chosen at least once or a ‘1’ for when you don’t

**Render as block**

In a survey using a ‘matrix’ style of organizing the items and responses (where there is a table with items down the left and response option across the top) sometimes single questions rendered in mResearch’s ‘vertical’ style can seem distracting. “Render as block” can set any items to render in a ‘horizontal’ style like matrixed items.

```xml
<item itemRef="7">
  <settings>
    <renderAsBlock>1</renderAsBlock>
  </settings>
</item>
```

**Series of text boxes that must equal a certain amount**

- This item type can allow percentages to add up to 100% with validation
- Other summations are also possible

```xml
<item itemRef="8">
  <itemType>6</itemType>
  <settings>
    <total>100</total>
    <tolerance>0</tolerance>
  </settings>
</item>
```

1. (Survey designer) Create a text item for each response option
2. (Survey designer) Insert a section break after the last item
3. (XML on the companion page) Identify the “itemref” of the first item
4. (XML on the companion page) Make this item a type ‘6’
5. (XML on the companion page) Use “total” to identify what all the numbers are supposed to add up to
6. (XML on the companion page) Use “tolerance” to identify the leeway you want give (a ‘tolerance’ of 5 in the above example would let the taker’s answers add up to 95 or 105 instead of 100)
Dual scale survey

〈item itemRef="12">
   〈itemType>7</itemType>
   〈item itemRef="15"/>
</item>

1. (Survey designer) In the survey designer of mResearch create the items that you want to share in a dual scale matrix (you will repeat these steps for each separate dual scale matrix that you want)
2. (Survey designer) Add items to the survey and repeat the items in the same sequence
3. (Survey designer) Give the first set the scale that will appear on the left and give the second set the scale that will appear on the right
4. (XML on the companion page) In the XML statement above change the first instance of itemRef to the Q number of the first item of the first scale and the second instance of itemRef to the Q number of the first item of the second scale
5. (XML on the companion page) Make the itemType = 7

The next four questions are in random order.

1. Each item to be randomized is identified and has it’s randomizeType set to 1
2. Each item must have the same scale
3. Response options can also be randomized and certain options can be used identified as anchors that don’t get randomized (like "other" or "I don’t know")

〈item itemRef="18">
   〈settings>
      〈randomizeType>1</randomizeType>
   </settings>
</item>

〈item itemRef="19">
   〈settings>
      〈randomizeType>1</randomizeType>
   </settings>
</item>

〈item itemRef="20">
   〈settings>
      〈randomizeType>1</randomizeType>
   </settings>
</item>

〈item itemRef="21">
   〈settings>
      〈randomizeType>1</randomizeType>
   </settings>
</item>
Filtering: Make an item disappear

1. Identify the value of the response options of the item that will drive the filtering
2. Use ‘itemRef’ to identify the Q number of the item that should disappear
3. Set the requirements that must be met for the filtering to occur
4. Like in branching the filtered item must be on a different page than the item or items identified as the requirement

<filtering>

  <item itemRef="23">
    <condition>
      <item itemRef="22" inverse="true">
        <response value="2" />
        <response value="3" />
      </item>
    </condition>
  </item>

  <item itemRef="24">
    <condition>
      <item itemRef="22" inverse="true">
        <response value="1" />
        <response value="3" />
      </item>
    </condition>
  </item>

  <item itemRef="25">
    <condition>
      <item itemRef="22" inverse="true">
        <response value="1" />
        <response value="2" />
      </item>
    </condition>
  </item>

</filtering>

Branching - Advanced features

1. Identify the item from which you want to branch
2. Set the item type to “basic”
3. Set what you want to happen for each value that can be chosen including; “none” for when no choice was made and “end” for when you want the taker to go to the end of the survey to submit the survey.

<branching>

  <item type="basic" itemRef="26">
    <response value="none" destinationItemRef="end" />
    <response value="1" destinationItemRef="end" />
    <response value="3" destinationItemRef="28" />
  </item>

</branching>
**Branching to the Finish**

The following code allows you to branch a respondent to the finish page by entering “finish” in the destination item ref. This is very similar to the previous branching to “end” but this new feature submits the survey and doesn’t allow the respondent to go back and change their answer.

```xml
<survey>
  <branching>
    <item itemRef="5" destinationItemRef="finish">
      <condition>
        <item itemRef="5">
          <response value="1" />
        </item>
      </condition>
    </item>
  </branching>
</survey>
```
Conditional Sets

You can refer to sets of conditions anywhere that needs to do condition checking. They can even be considered together (within a given container) or referenced from other condition nodes (as in the example below).

In the example below, at **item 2** you will branch to 5 if **item 1** is 1 or 2 OR if **item 2** is 3.

Then at **item 5**, you will branch to the end if both **item 1** is 1 or 2 AND **item 2** is 3 AND **item 5** is 1.

You can mix and match both condition sets and inline condition nodes.

```
<survey>
  <conditionSets>
    <EmploymentQualifier>
      <condition>
        <item itemRef="1">
          <response value="1" />
          <response value="2" />
        </item>
      </condition>
    </EmploymentQualifier>
    <EmploymentQualifierB>
      <condition>
        <item itemRef="2">
          <response value="3" />
        </item>
      </condition>
    </EmploymentQualifierB>
    <SuperEmploymentQualifier operator="or">
      <condition conditionSet="EmploymentQualifier" />
      <condition conditionSet="EmploymentQualifierB" />
    </SuperEmploymentQualifier>
  </conditionSets>
  <branching>
    <item itemRef="2" destinationItemRef="5" >
      <condition conditionSet="SuperEmploymentQualifier" />
    </item>
    <item itemRef="5" destinationItemRef="end" operator="and" >
      <condition conditionSet="EmploymentQualifier" />
      <condition conditionSet="EmploymentQualifierB" />
      <condition>
        <item itemRef="5">
          <response value="1" />
        </item>
      </condition>
    </item>
  </branching>
</survey>
```
Conditional Redirection
This code allows you to conditionally redirect a user. In this example, if someone selected value=10 on item 6 they would be redirected to www.modernsurvey.com after they hit submit survey.

```xml
<survey>
  <settings>
    <finishText>
      <conditions>
        <condition>
          <item itemRef="6">
            <response value="10" />
          </item>
        </condition>
      </conditions>
      <value>
        <meta http-equiv="refresh" content="0;url=https://www.modernsurvey.com">
      </value>
    </finishText>
  </settings>
</survey>
```

Survey Finish Time
The following code checks if the respondent completed the survey in less than 120 seconds. If it's true it returns a custom finish text and/or has the ability to redirect to a special location.

```xml
<survey>
  <settings>
    <finishText>
      <conditions>
        <condition>
          <customer>
            <secondonsurvey comparison="lessthan">120</secondonsurvey>
          </customer>
        </condition>
      </conditions>
      <value>
        <h3>You finished too fast</h3>
      </value>
    </finishText>
  </settings>
</survey>
```
Mobile Friendly Selectors
Slider buttons are commonly used to make surveys easier to use on mobile devices.

1. Identify the block of items from which you want to apply this format
2. The itemRef number represents the question number to start the friendly selectors. In this example, the xml will change all the selectors for this question type. If there is a section break it's necessary to identify where the selector type should start again.
   a. Questions 1-12 are a 5 point scale
   b. Questions 13-15 are Yes/No questions
   c. There is a section break after question 15
   d. Questions 16-20 are a 5 point scale

   <survey>
   <items>
     <item itemRef="1">
       <itemType>RatingSlider</itemType>
     </item>
     <item itemRef="13">
       <itemType>RatingSlider</itemType>
     </item>
     <item itemRef="16">
       <itemType>RatingSlider</itemType>
     </item>
   </items>
   </survey>

Survey Theme
New survey’s created have a theme by default. To remove this theme and make your surveys look generic change the Survey Theme to Original from the drop down menu.