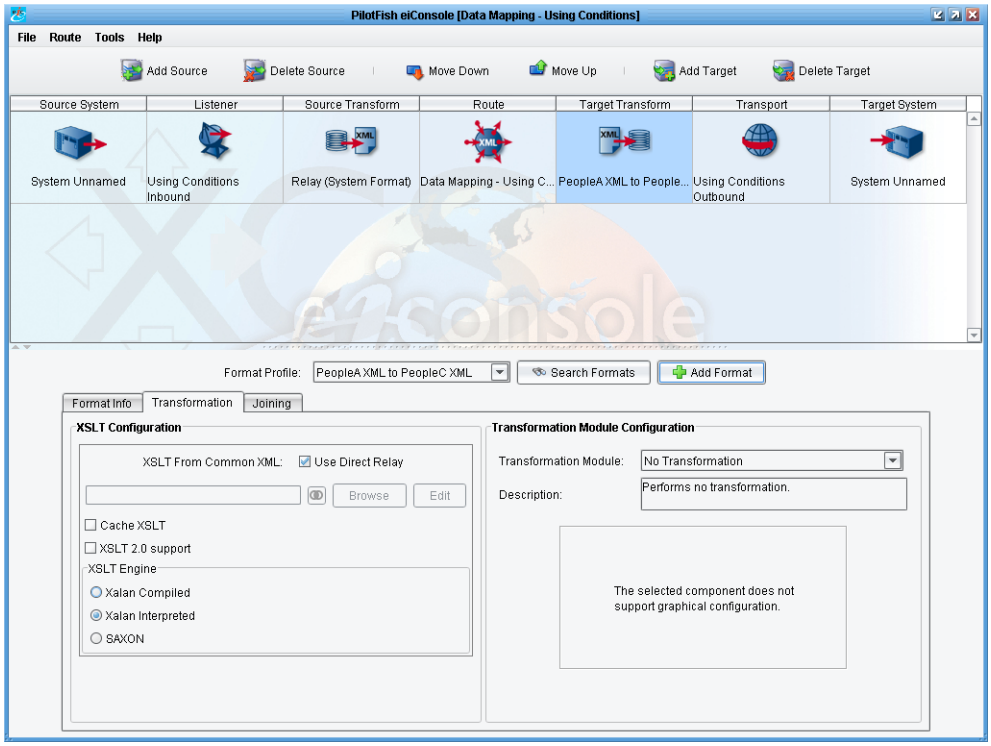
Data Mapping – Using Conditions

Overview

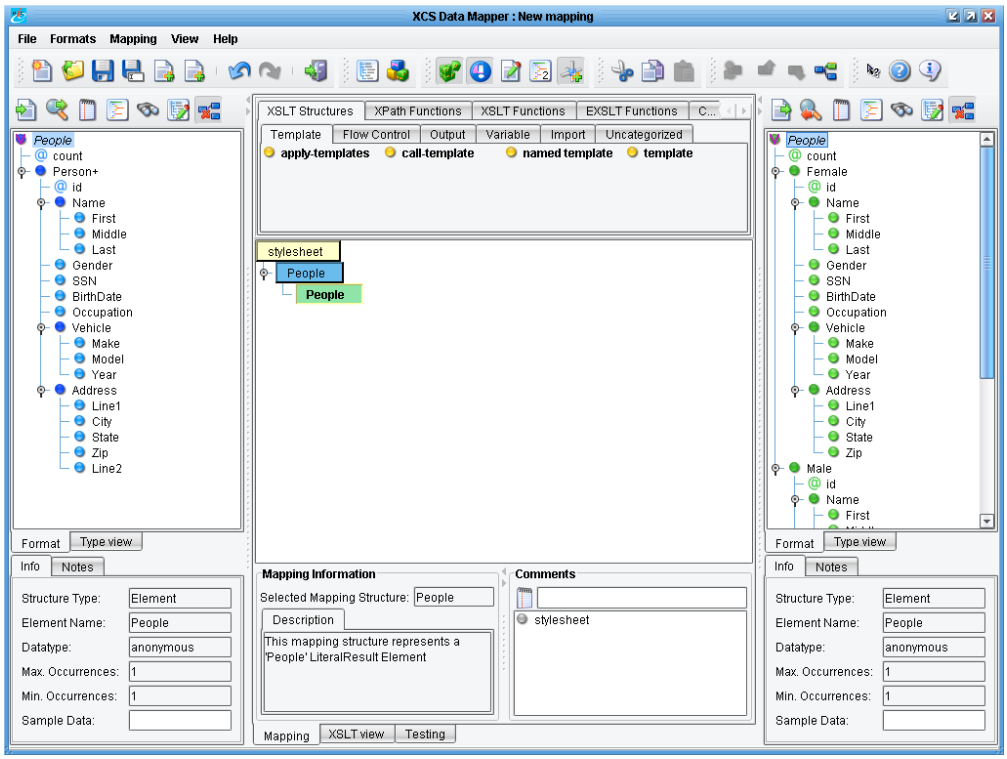
In this tutorial we'll cover the use of XSLT's conditional statements to map Person elements to "Male" or "Female" elements based on "Gender." This tutorial expands on concepts covered in "Data Mapping – Using Iteration," so users are expected to be familiar with that material.

Steps

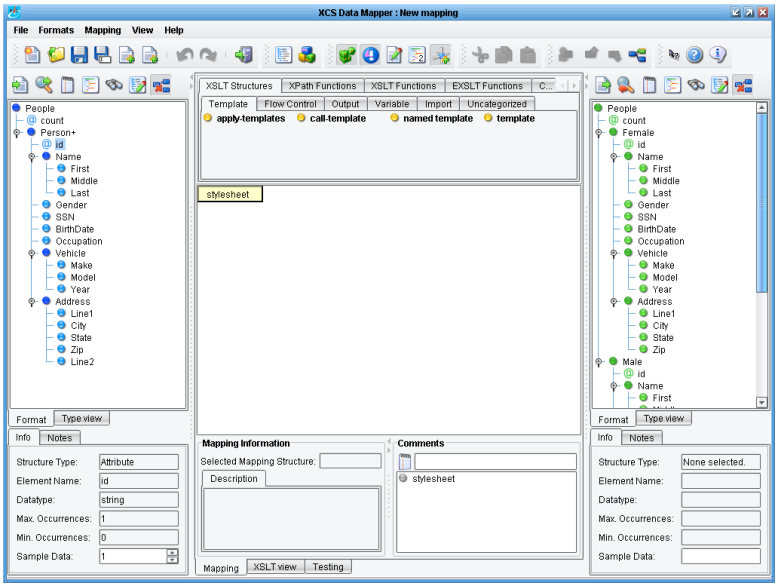
Start by creating and configuring a new Route following the same steps as the previous tutorial. Add a new Format named "PeopleA to PeopleC":



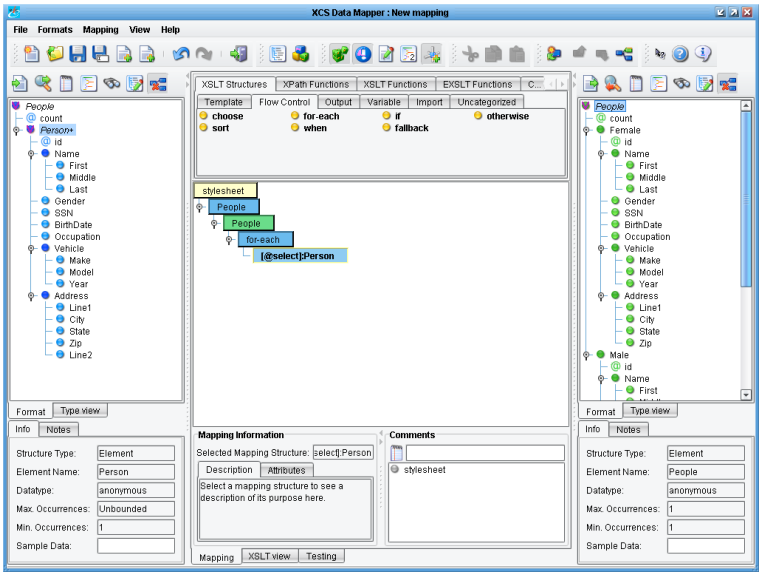
Uncheck "Use Direct Relay" and click "New" to open the Data Mapper. Using the XML Format Reader, read "PeopleA.xml" as the Source and "PeopleC.xml" as the Target:



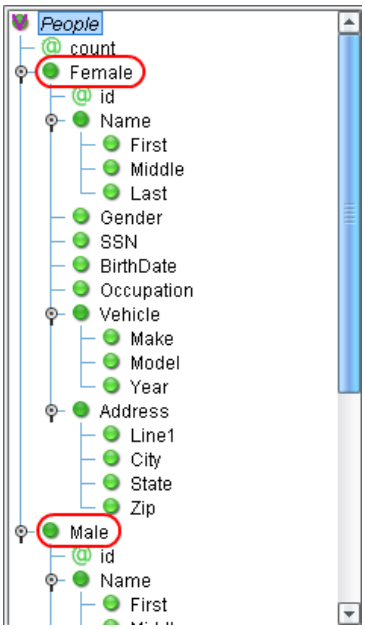
Begin by mapping "People" from the Source onto the "stylesheet" element and "People" from the Target onto "People" in the center:



Using XSLT Structures → Flow Control, drag a "for-each" instruction onto the People element from the Target in the panel, then drag "Person" from the Source onto its "@select" attribute:

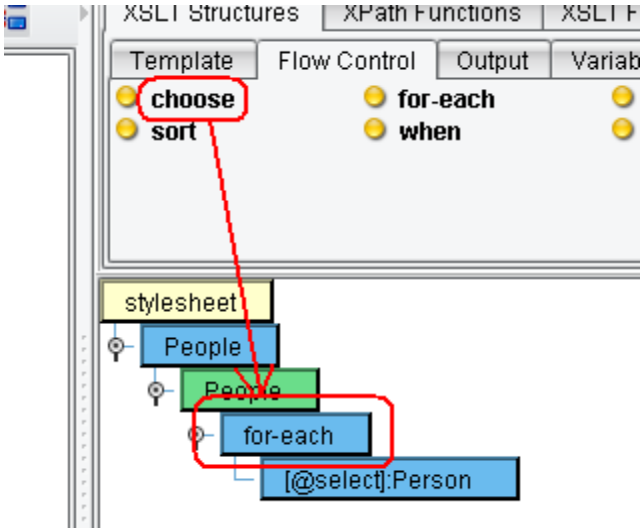


If you review the Target format, you'll see that instead of the "Person" elements from before, we instead have "Female" and "Male" elements. The child elements these are the same, but their names differ:

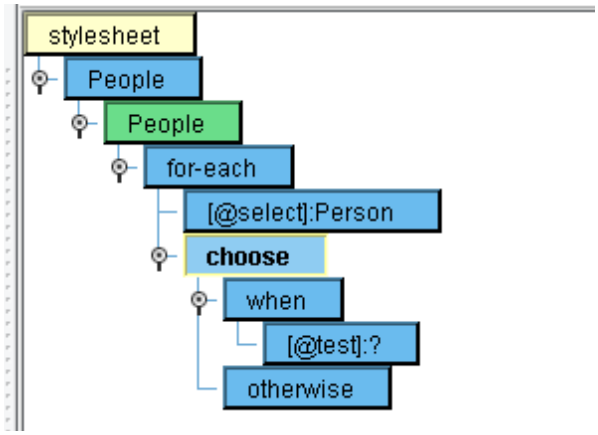


We'll want to conditionally create either a Male or a Female element based on a given Person's "Gender" from the Source. There are a variety of ways to do this, but we'll use the simplest and most straightforward: XSLT's choose / when / otherwise instructions.

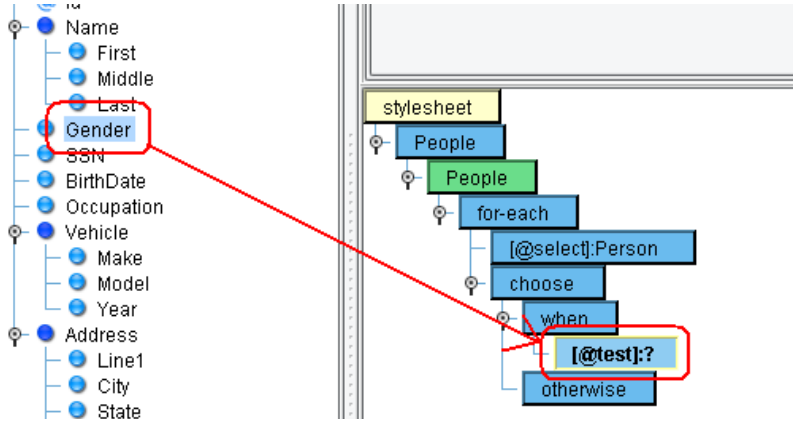
XSLT has three ways of handling conditions: the "if" instruction, the "choose," "when," and "otherwise" instructions, and predicates ("where" clauses on expressions). There is no "else" or "else if" in XSLT, so "if" is only useful for evaluating single expressions unless each subsequent expression has the added complication of testing its prior "if" instruction's expression, as well. For this tutorial, we'll use the "choose" method. Drag "choose" from the XSLT Structures → Flow Control palette menu onto the "for-each" element:



This will create a "choose" element in the mapping with a single "when" element and an "otherwise" element:



A "choose" instruction can have one or more "when" elements; if you require more, simply drag the "when" palette item onto an existing "choose." Our mapping only makes use of a single case, however; we'll test if Gender is "Female" and use the "otherwise" to default to "Male." Drag "Gender" from the Source onto the "when" instruction's "@test" attribute:

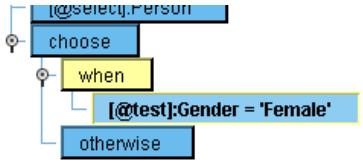


This will establish the basis of our test expression. However, as-is, "@test" will have only the value of Gender for its expression, which isn't a particularly useful test. We'll need to modify it. Double-click on "@test" in the mapping to edit it:

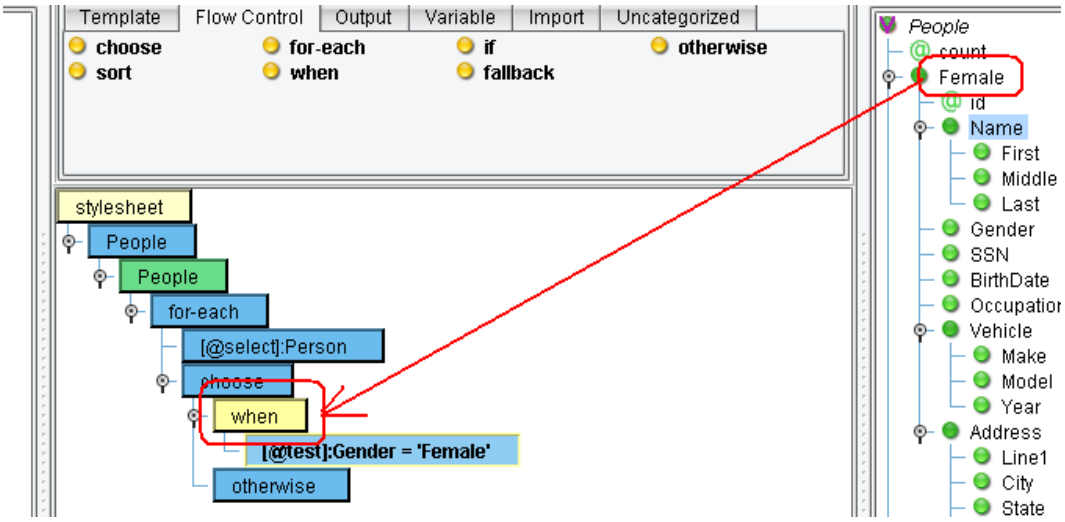
You can use the "…" button to open an XPath Expression Editor, but for this case, we'll simply type our expression in. Modify it to read:

Gender = 'Female'

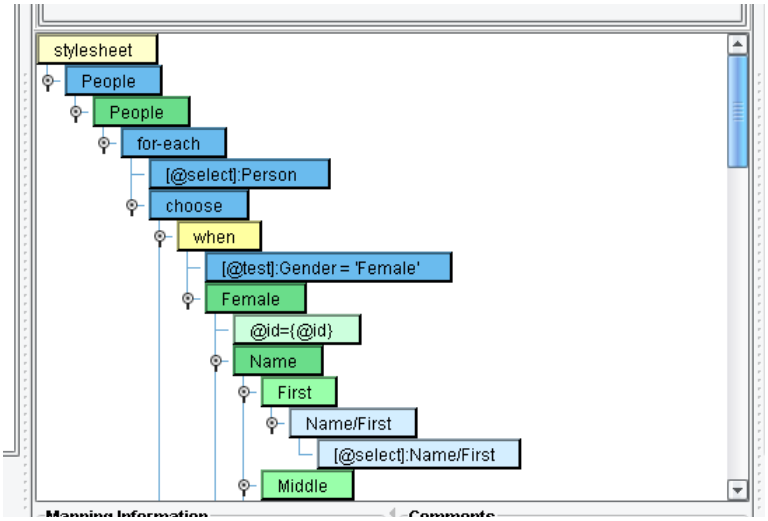
Then hit enter to save the changes. Clicking elsewhere will revert the expression to its previous value:



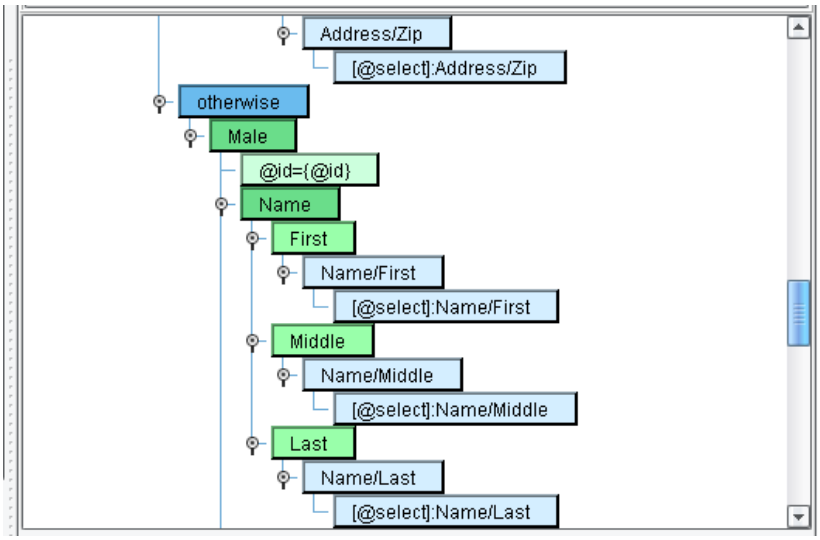
Next, drag the "Female" element from the Target onto "when":



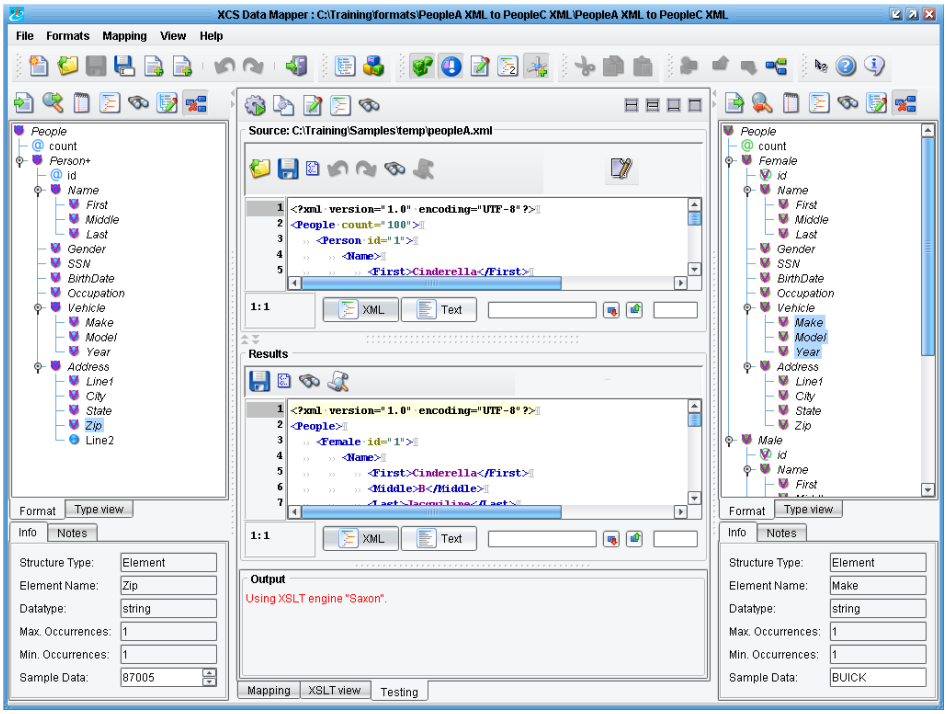
Map the Female element's child elements, then map the Person element in the Source's corresponding elements onto the Female element in the center's:



Map "Male" from the Target onto "otherwise" and repeat the mapping exercise (Source to Mapping):



Finally, test the mapping:



If you receive one of the following errors make sure that your output has ONE root element:

- Using XSLT engine "Xalan (Interpreted)".
  [Fatal Error] java.util.EmptyStackException
  Transformation error : java.util.EmptyStackException

- Using XSLT engine "Xalan (Compiled)".
  Transformation error : org.w3c.dom.DOMException:
  HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted.

- Using XSLT engine "Saxon".
  [Fatal Error] org.w3c.dom.DOMException: HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted. ; SystemID: ; Line#: 8; Column#: -1
  Transformation error : org.w3c.dom.DOMException:
  HIERARCHY_REQUEST_ERR: An attempt was made to insert a node where it is not permitted.

*** Bonus ***

Assuming that order is important; generate all <Female> aggregates first followed by <Male> aggregates. Hint: use <xsl:sort> in the <xsl:for-each>.