

In this course you will be using <oxygen/> XML Editor version 14 (oxygen for short from now on) for XML related work. The work includes writing XML Schema files with corresponding XML files, writing XQuery code and running it.

We will now go through the above listed tasks and see how they're done in oxygen. This is not a tutorial about XML and XML related technologies in themselves, but rather how you work with the things just mentioned in oxygen.

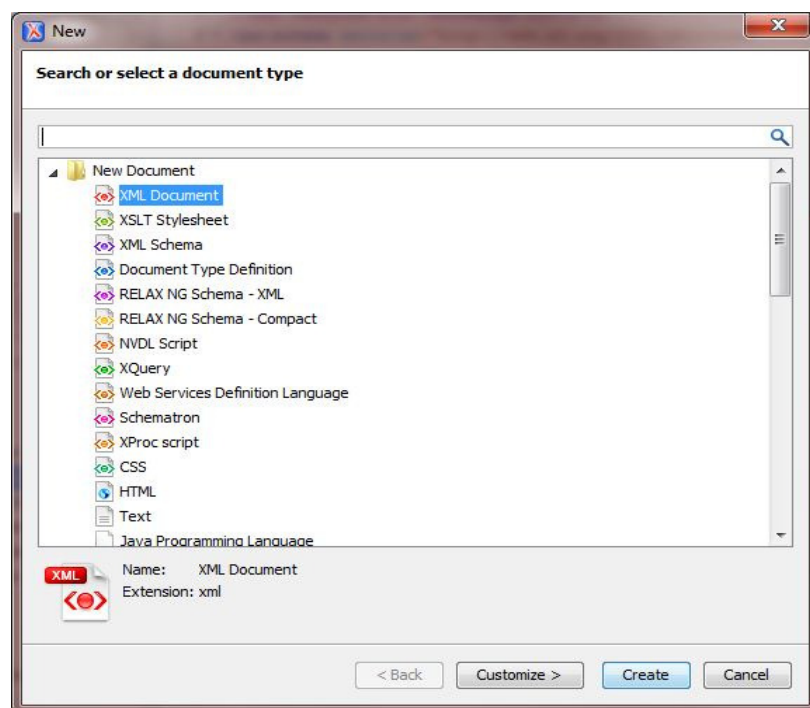
If this is your first time working with XML it might be a good idea to try and follow the tutorial actively and not just simply reading or skimming through it.

# 1. XML Schemas and XML data files

Writing XML Schemas and XML data files is a pleasant task in oXygen compared to a dumb text editor like Notepad for instance. Not only does oXygen assist with automatic syntax coloring and indentation, it also provides useful auto-completions, suggestions etcetera.

## 1.1 Create XML Schema

To create a new XML Schema or XML file you simply select the menu item "File->New..." which brings up a dialog (picture below) where you can choose the appropriate type. From this dialog you can also create files for XQuery (these are all raw text files so, basically, you could do everything manually if you wanted, but oXygen can assist with skeleton content if you use the dialog).



Choose "XML schema" to create your schema file. A graphical representation of the schema will appear in oXygen. Click the "Text" button at the bottom of the editor and you should end up with a skeleton (see below) which enables you to start defining the specific structure of the schema right away.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

</xs:schema>
```

Now you can write your code or schema. In the following we will use the simple schema bellow saved as cars.xsd:

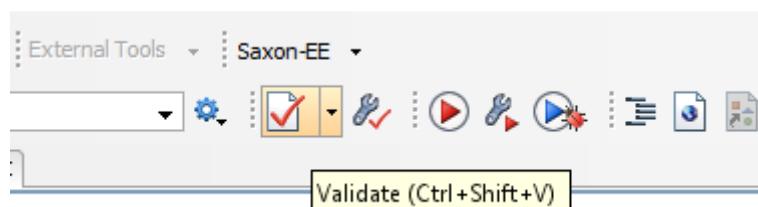
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="cars" type="rootType"/>

  <xs:complexType name="rootType">
    <xs:sequence>
      <xs:element name="car" type="carType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="carType">
    <xs:sequence>
      <xs:element name="maker" type="xs:string"/>
      <xs:element name="model" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

## 1.2 Validate Schema

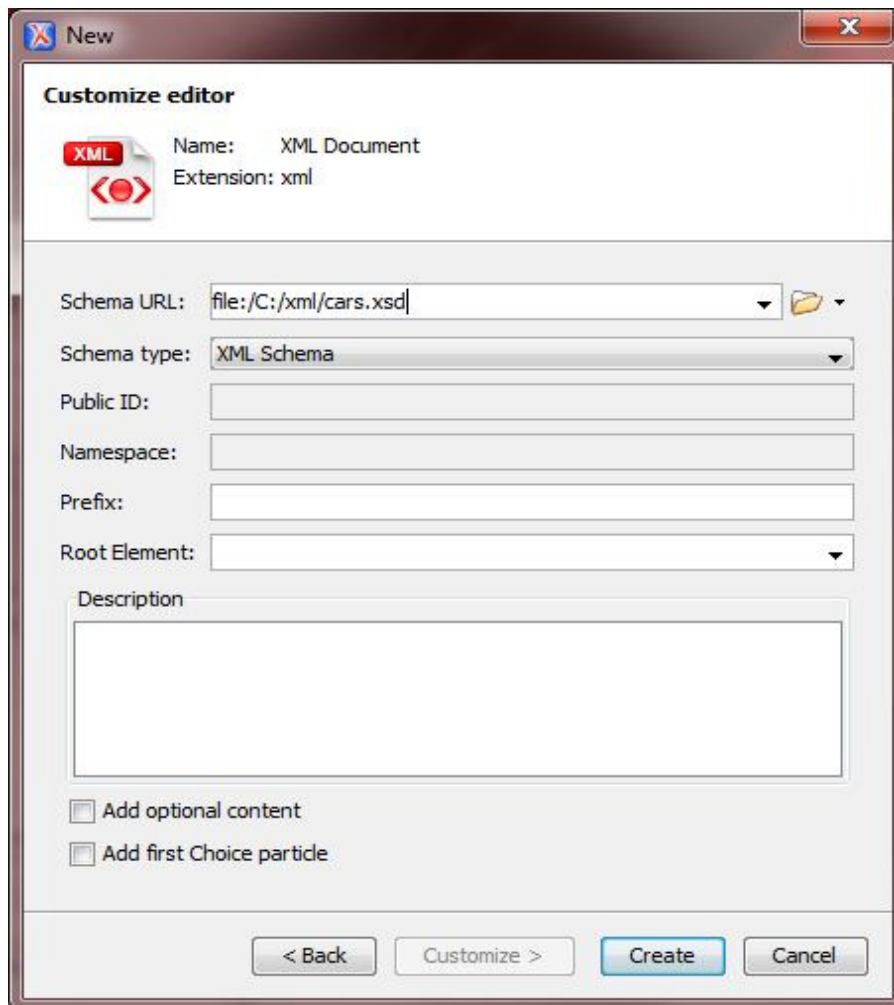
If you want to validate this schema you can press the toolbar button "Validate" (displayed below) or access the menu item "Document->Validate->Validate" or press Ctrl+Shift+V. However, oXygen performs on-the-fly validation so you won't need to do it manually very often. If you have several files open (oXygen can have many files open in tabs), the correct one must have focus.



Any validation errors will be underlined in the document itself and when you press the "Validate" button a window will appear in the bottom area with a summary of all errors (if there were any).

### 1.3 Create XML File

Now when we have a simple but complete schema and can create a corresponding XML file. Using the dialog "New" described earlier we create an XML file. By clicking the "Customize >" button in the window above we can specify a schema for the document, which is convenient, because our skeleton will be more complete. The dialog below illustrates how to perform the schema association.



Below is what the skeleton will look like if we create a new XML file with the cars.xsd-schema associated.

```
<?xml version="1.0" encoding="UTF-8"?>
<cars xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="file:/C:/oxygen-tutorial/cars.xsd">
  <car>
    <maker></maker>
    <model></model>
  </car>
</cars>
```

This skeleton is now ready to be filled with data and you will notice that oXygen knows the order of the elements which enables it to perform a lot of auto-completions.

However, as you can see the file URL is specified using an absolute path which is not very convenient if you move the files between different computers and/or users. If we save this file as cars.xml and keep it together with the schema we can change the URL to:

```
xsi:noNamespaceSchemaLocation="file:cars.xsd"
```

If you were using namespaces you would be setting `xsi:schemaLocation` instead and the syntax would be slightly different.

## 2. XQuery

We continue on our cars example, say the file cars.xml now contains:

```
<?xml version="1.0" encoding="UTF-8"?>
<cars xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="file:cars.xsd">
  <car>
    <maker>BMW</maker>
    <model>750Li</model>
  </car>

  <car>
    <maker>BMW</maker>
    <model>M5</model>
  </car>
  <car>
    <maker>Volvo</maker>
    <model>V70</model>
  </car>
  <car>
    <maker>Ford</maker>
    <model>F150</model>
  </car>
</cars>
```

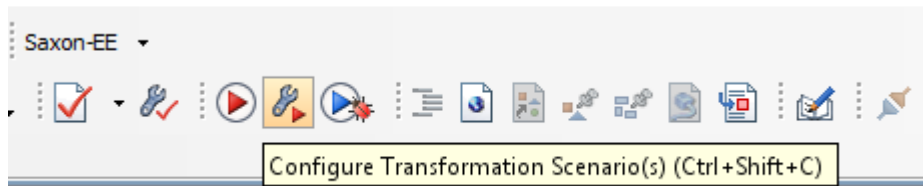
### 2.1 Create XQuery

We will write an XQuery that returns how many cars there are where the maker is BMW. To create an xquery-file you use the now familiar dialog "New". This will give you an Untitled file with the extension xquery that is empty. Save it as num\_bmw.xquery in the same directory as cars.xsd and cars.xml. Write the code:

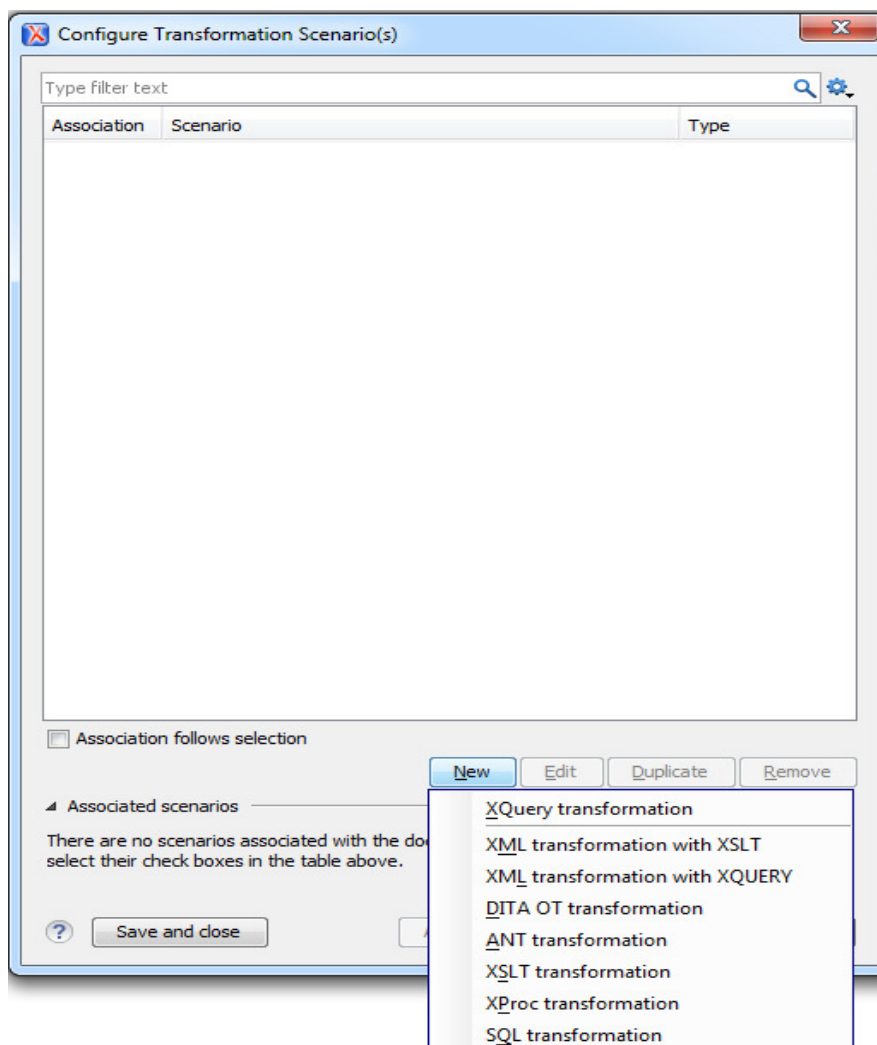
```
<res>Num cars by BMW: {count(doc('cars.xml')//car[maker='BMW'])}</res>
```

## 2.2 Configure oXygen for XQuery Execution

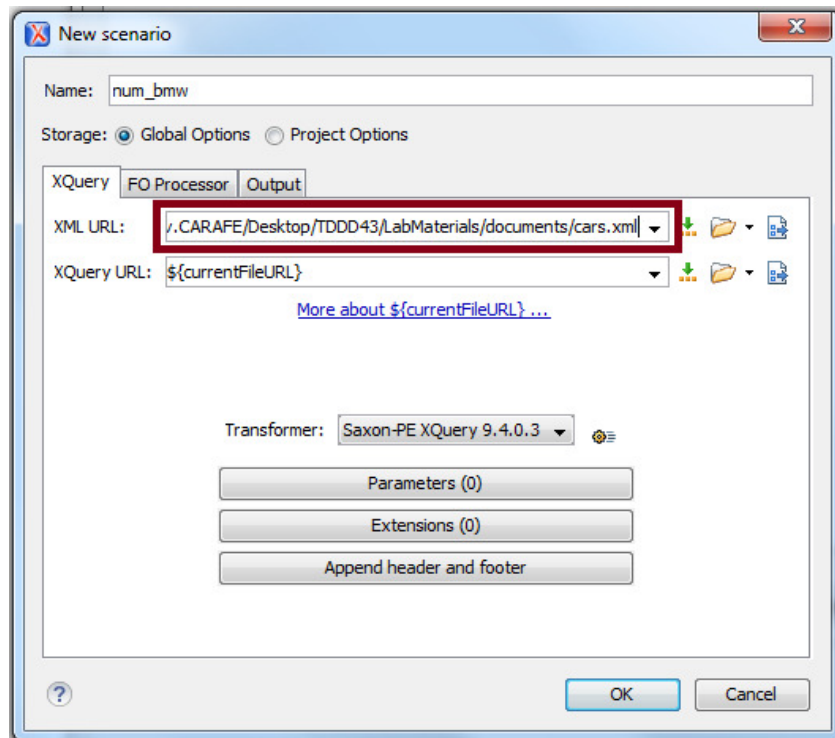
Now that we have the code for the query, it's time to run it. Make sure the XQuery file has focus and press the button "Configure Transformation Scenario(s)".



The "Configure Transformation Scenario(s)" dialog appears.



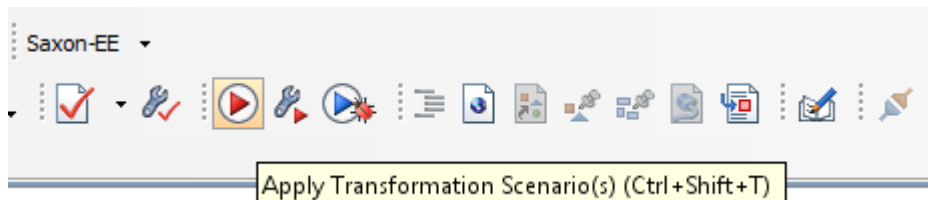
Press the "New" button and choose the "XQuery transformation" option. Then the following dialog will appear.



Enter the path to your file with XML data and press “OK”.

## 2.3 Run XQuery

Now that the transformation scenario has been configured, it's time to run it. Press the “Apply Transformation Scenario(s)” button:



When you press it the XQuery should be executed and the result displayed at the bottom:

```
<?xml version="1.0" encoding="UTF-8"?>  
<res>Num cars by BMW: 2</res>
```



## 4. A few tips...

If you want you can create so called projects in oXygen where you can group related files. You don't have to work using oXygen projects in the course but some might find it easier to organize things that way.

To toggle comments on the current selection: Ctrl + Shift + ,

That key combination is very useful indeed!

Most of the time oXygen indents for you automatically, but certain times you have to indent blocks of text manually. The key combination for that is: Ctrl + i

However, sometimes the indentation mechanism refuses to do its work when Ctrl + i is pressed and then selecting the same thing from the menu usually works. That menu item can be found at "Document->Source->Indent Selection".

This was just a very short introduction of how oXygen works to get you started on the labs, explore the program yourselves!