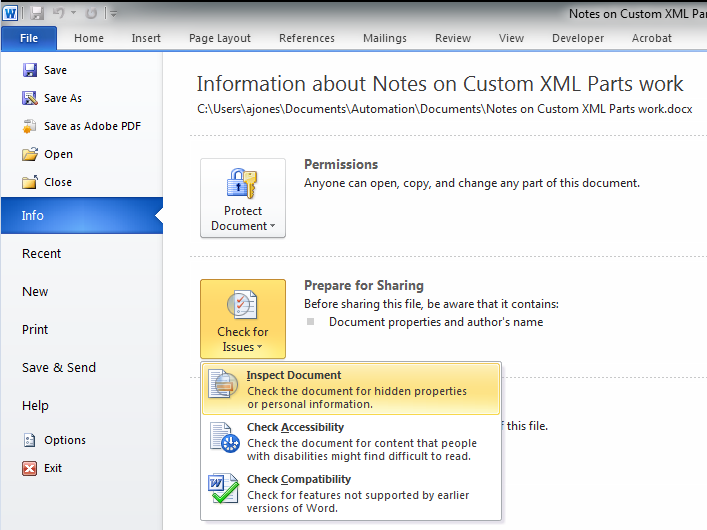
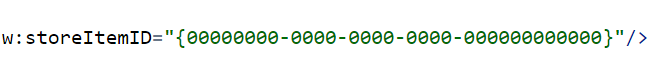
**Notes on Custom XML Parts work**

March 21, 2017

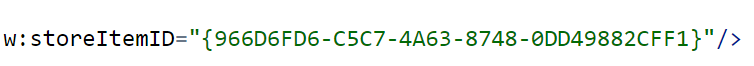
A custom XML part can be *manually* removed:



The text and the content controls remain, but underneath in the Open XML you get this:



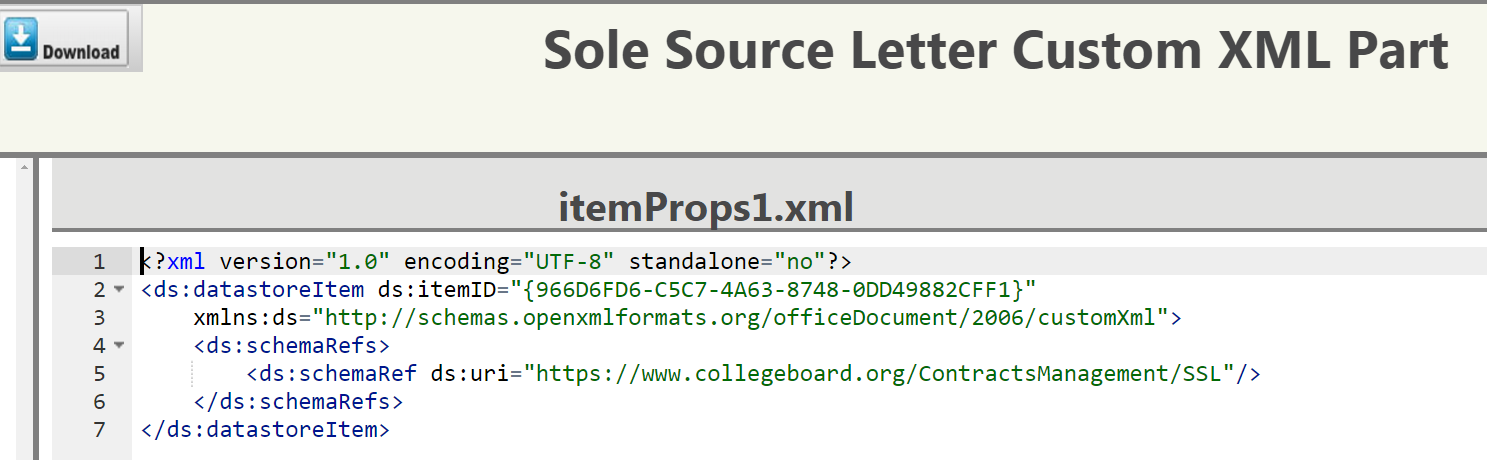
What it used to be:



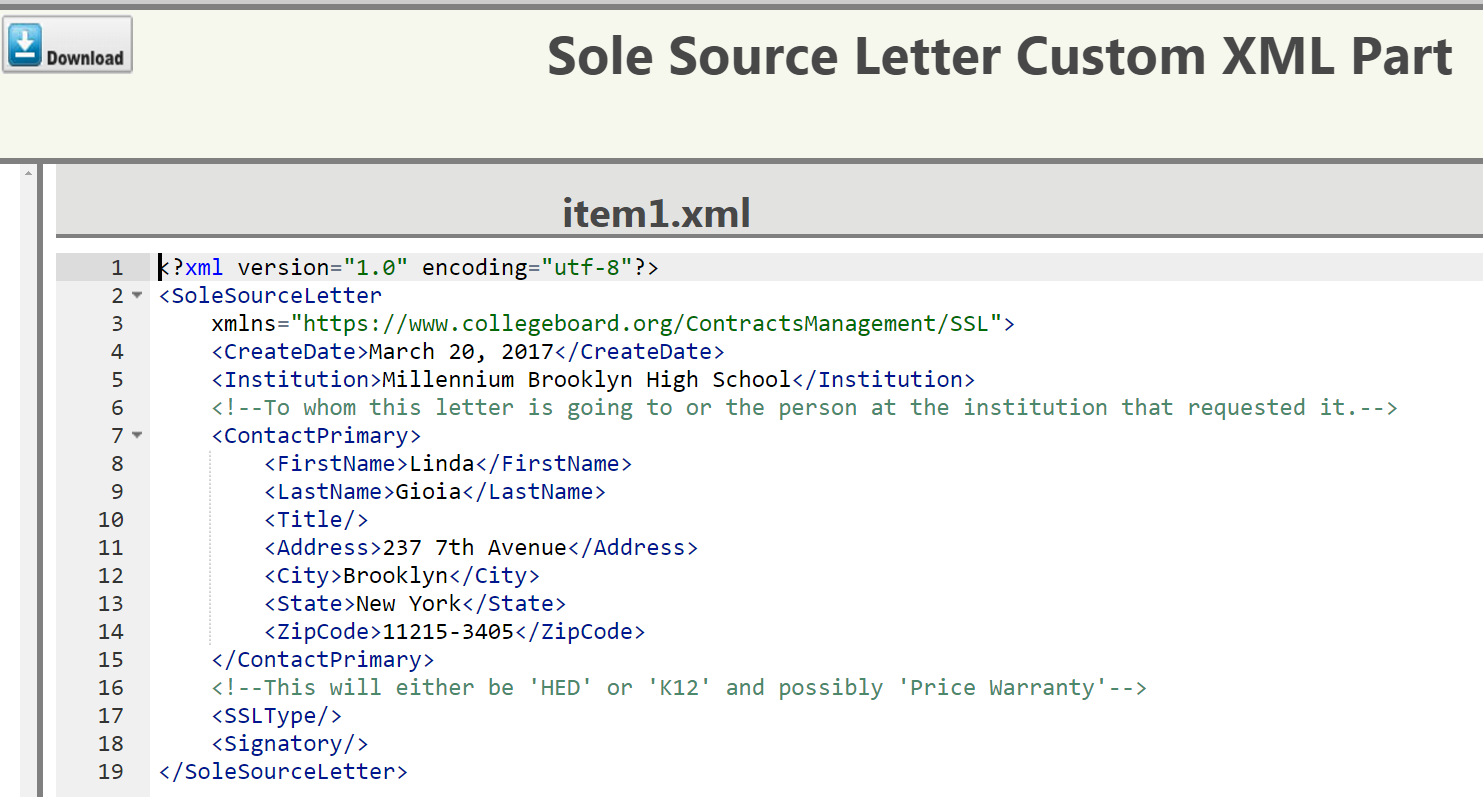
In the Custom XML “Folder” there’s:



Props contains what’s below:



And item1.xml is the actual XML file with the data:



Wednesday, March 22, 2017

My goal is to have a strong understanding of Custom Xml Parts. So I’m starting with this:

|  |  |
| --- | --- |
| When looking in the Custom XML Part in a document | The Open XML SDK’s expression of a Custom XML Part |
|  |  |

From my understanding the property *RelationshipType* corresponds to the *..rels* folder you see in the document. The property *CustomXmlPropertiesPart* corresponds to the *itemProps1.xml* file. Now that leaves the *item1.xml* file/part(?) that I want to access, but I’m unsure how. It has a base type of OpenXmlPart so maybe I should think of approaching it in that capacity?

**COPY THE CODE THAT MAKES IT WORK HERE!!**

foreach (CustomXmlPart currCXP in SSLMain.CustomXmlParts)

{

Console.WriteLine("Uri: {0}", currCXP.Uri.ToString());

Console.WriteLine("Child Part (CustomXmlProperties) Uri: {0}", currCXP.CustomXmlPropertiesPart.Uri.ToString());

Console.WriteLine("Child Part (Relationship) {0}", currCXP.RelationshipType);

Console.WriteLine("DataStoreItem: {0}", currCXP.CustomXmlPropertiesPart.DataStoreItem.ItemId);

Console.WriteLine();

/\* Get the stream of the XML File sample

\* THIS WORKS!!!!

\*/

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

StreamReader strdXML = new StreamReader(strFULLPATH\_SSL\_XML\_SAMPLE);

currCXP.FeedData(strdXML.BaseStream);

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

}

Console.ReadLine();

SSLDoc.MainDocumentPart.Document.Save();

**Creating a Template of Linked Content Controls.**

March 24, 2017

Interesting note: Since Word 2007, the *default* for inserting document properties into a Word document has been expressed as a content control instead of a field:

|  |  |
| --- | --- |
| **Document Property - Content Control** *(The default)* | |
|  | Author Document Property inserted: Adrian Jones |

|  |  |
| --- | --- |
| **Document Property - Field** | |
|  | Author Document Property inserted: |

March 27, 2017

Okay, the **IMPORTANT** **NOTE** below is followed by the code *needed* to retrieve the *name* I give any of these XML files

**IMPORTANT!:** What’s important to remember is I’M RESPONSIBLE FOR WHAT THE ABOVE (the custom xml file) IS CALLED then I should LOOK for what I CALLED IT! DUH!! (see code below)

//Read the XML from the Custom Part by using its uri

XDocument CM\_CustomXML = new XDocument();

CM\_CustomXML = XDocument.Load(currCXP.GetStream());

XElement CM\_RootCustomXML = new XElement(CM\_CustomXML.Root);

if (CM\_RootCustomXML.Name.LocalName != string.Empty)

{

Console.WriteLine("The Root ==> {0}", CM\_RootCustomXML.Name.LocalName);

}

Alright then, what’s the name for the contract xml file? Contracts?

Git Repositories

1. OperationKyuzo
   1. Working with Custom XML Parts
      1. **Branch Name:** CM\_XMLParts
   2. Working with Finding and Inserting AutoText
      1. **Branch Name:** CM\_AutoText
   3. Sole Source Letter
      1. **Branch Name:** CM\_SSL
   4. Contracts – Standards
      1. **Branch Name:** CM\_Contracts