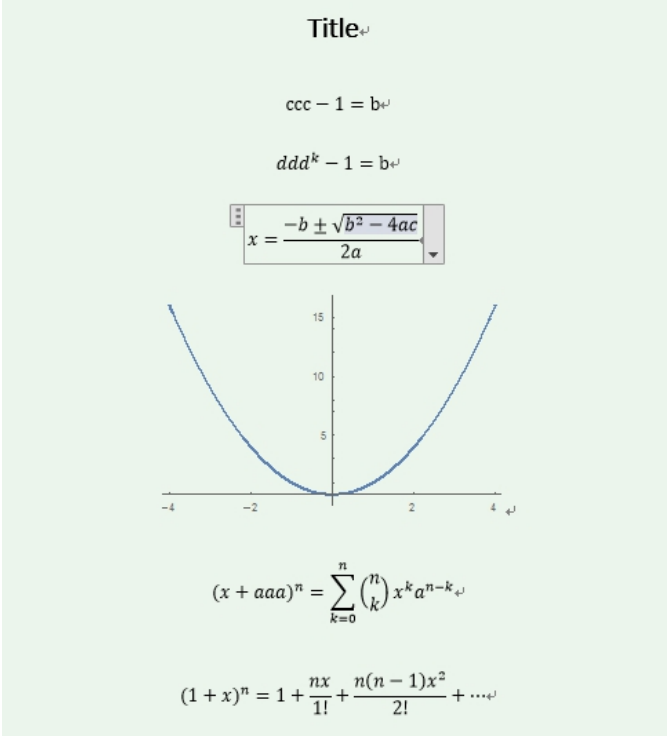
# How to programmatically extract mathematical formulas from Word with Mathematica 9?

I have **a bunch of(1000+)** Microsoft Word document in .Docx format. How can I programmatically extract the mathematical formulas from MS Word using *Mathematica* 9?

This is what example looks like (or download link) :




I would really appreciate if someone has an answer to my question!

import external-calls office

edited Dec 4 '15 at 13:19
xzczd
17.6k 2 41 149

asked Nov 14 '14 at 2:48
Xiang Li
140 1 7

What do you mean by "How to get"? How to copy/paste? How to read it from a doc file? Other? – Dr. belisarius Nov 14 '14 at 3:01

Hi, @belisarius. I'd like to read formula Programmatically from doc file. – Xiang Li Nov 14 '14 at 3:17

This may help mathematica.stackexchange.com/q/27406/193 – Dr. belisarius Nov 14 '14 at 4:28

3 If you use the new style equation editor in Word 2007 and later then you can simply copy and paste formulae to Mathematica. It uses MathML as the exchange format. – Szabolcs Nov 14 '14 at 5:03

1 @Szabolcs Thanks for sharing your idea. I updated the C# code to return MathML. I dont have Mathematica on this PC but will try it tonight. Fingers crossed!!! – WolframFan Nov 17 '14 at 0:55

## 1 Answer

Get all the files here: http://JeremyThompson.net/Rocks/Mathematica/MmaWord.zip

## .Net Mathematica Word Library

You will need to use a Microsoft library to open word documents. In a language such as .Net it is very easy; just open Visual Studio, reference the Microsoft.Office.Interop.Word .Net DLL (for Words) and the C:\Program Files\Open XML SDK\V2.5\lib\DocumentFormat.OpenXml.dll (for Formulas in the MathML format). Then you build this C# code:

```
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Xml;
using System.Xml.Xsl;
using DocumentFormat.OpenXml.Packaging;
using Microsoft.Office.Interop.Word;

namespace MathematicaWordHelper
{
    public class WordHelper
    {
        /// <summary>
        /// Opens a Microsoft Word Document and returns the content of words
        /// </summary>
        /// <param name="docFilePath"></param>
        /// <returns></returns>
        public string GetWordDocumentText(string docFilePath)
        {
            string output = string.Empty;

            // Open word
            _Application oWord = new Application();
            _Document oDoc = oWord.Documents.Open(docFilePath, ReadOnly: true);

            // Get the Documents text
            output = oDoc.Content.Text.ToString();

            // Close word
            oDoc.Close();
            oWord.Quit(false);
            Marshal.ReleaseComObject(oDoc);
            Marshal.ReleaseComObject(oWord);

            // Return the text to Mathematica calling code
            return output;
        }

        /// <summary>
        /// This is an overloaded method for ease of use (on most PCs where MS Word is
installed in the default location)
        /// </summary>
        /// <param name="docFilePath"></param>
        /// <param name="officeVersion"></param>
        /// <returns></returns>
        public string GetWordDocumentAsMathML(string docFilePath, int officeVersion = 15)
        {
            return GetWordDocumentAsMathML(docFilePath,
                                    @"c:\Program Files\Microsoft Office\Office" +
officeVersion.ToString() +
                                    @"\OMML2MML.XSL");
        }

        /// <summary>
        /// This returns one formula of all the Equations in a Microsoft Document in Math
ML format, ref: http://en.wikipedia.org/wiki/MathML
        /// </summary>
        /// <param name="docFilePath"></param>
        /// <param name="officeMathMLSchemaFilePath"></param>
        /// <returns></returns>
        public string GetWordDocumentAsMathML(string docFilePath, string
officeMathMLSchemaFilePath = @"c:\Program Files\Microsoft Office\Office15\OMML2MML.XSL")
        {
            string officeMLFormulaAllTogether = string.Empty;
            using (WordprocessingDocument doc = WordprocessingDocument.Open(docFilePath,
false))
            {
                string wordDocXml = doc.MainDocumentPart.Document.OuterXml;

                XslCompiledTransform xslTransform = new XslCompiledTransform();
                xslTransform.Load(officeMathMLSchemaFilePath);

                using (TextReader tr = new StringReader(wordDocXml))
                {
                    // Load the xml of your main document part.
                    using (XmlReader reader = XmlReader.Create(tr))
                    {
                        using (MemoryStream ms = new MemoryStream())
                        {
                            XmlWriterSettings settings =
xslTransform.OutputSettings.Clone();

                            // Configure xml writer to omit xml declaration.
```

```
                            settings.ConformanceLevel = ConformanceLevel.Fragment;
                            settings.OmitXmlDeclaration = true;

                            XmlWriter xw = XmlWriter.Create(ms, settings);

                            // Transform our OfficeMathML to MathML.
                            xslTransform.Transform(reader, xw);
                            ms.Seek(0, SeekOrigin.Begin);

                            using (StreamReader sr = new StreamReader(ms, Encoding.UTF8))
                            {
                                officeMLFormulaAllTogether = sr.ReadToEnd();
                            }
                        }
                    }
                }
            }
            return officeMLFormulaAllTogether;
        }

        /// <summary>
        /// This is an overloaded method for ease of use (on most PCs where MS Word is
installed in the default location)
        /// </summary>
        /// <param name="docFilePath"></param>
        /// <param name="officeVersion"></param>
        /// <returns></returns>
        public string[] GetWordDocumentAsMathMLFormulas(string docFilePath, int
officeVersion = 15)
        {
            return GetWordDocumentAsMathMLFormulas(docFilePath,
                                    @"c:\Program Files\Microsoft Office\Office" +
officeVersion.ToString() +
                                    @"\OMML2MML.XSL");
        }

        /// <summary>
        /// This returns a string array of all the separate Equations in a Microsoft
Document in Math ML format, ref: http://en.wikipedia.org/wiki/MathML
        /// </summary>
        /// <param name="docFilePath"></param>
        /// <param name="officeMathMLSchemaFilePath"></param>
        /// <returns></returns>
        public string[] GetWordDocumentAsMathMLFormulas(string docFilePath, string
officeMathMLSchemaFilePath = @"c:\Program Files\Microsoft Office\Office15\OMML2MML.XSL")
        {
            List<string> officeMLFormulas = new List<string>();
            using (WordprocessingDocument doc = WordprocessingDocument.Open(docFilePath,
false))
            {
                foreach (var formula in
doc.MainDocumentPart.Document.Descendants<DocumentFormat.OpenXml.Math.Paragraph>())
                {
                    string wordDocXml = formula.OuterXml;

                    XslCompiledTransform xslTransform = new XslCompiledTransform();
                    xslTransform.Load(officeMathMLSchemaFilePath);

                    using (TextReader tr = new StringReader(wordDocXml))
                    {
                        // Load the xml of your main document part.
                        using (XmlReader reader = XmlReader.Create(tr))
                        {
                            using (MemoryStream ms = new MemoryStream())
                            {
                                XmlWriterSettings settings =
xslTransform.OutputSettings.Clone();

                                // Configure xml writer to omit xml declaration.
                                settings.ConformanceLevel = ConformanceLevel.Fragment;
                                settings.OmitXmlDeclaration = true;

                                XmlWriter xw = XmlWriter.Create(ms, settings);

                                // Transform our OfficeMathML to MathML.
                                xslTransform.Transform(reader, xw);
                                ms.Seek(0, SeekOrigin.Begin);

                                using (StreamReader sr = new StreamReader(ms,
Encoding.UTF8))
                                {
                                    officeMLFormulas.Add(sr.ReadToEnd());
                                }
                            }
                        }
                    }
                }
            }
            return officeMLFormulas.ToArray();
        }
    }
}
```

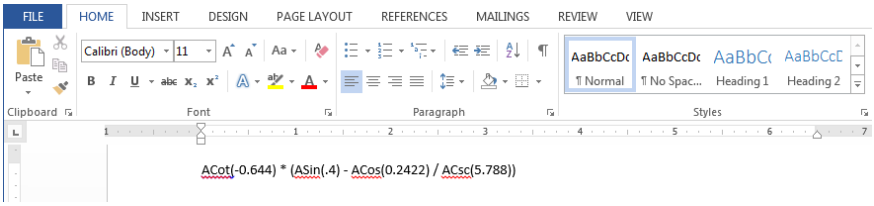
## Calling .Net from Mathematica

In a Mathematica NoteBook (or etc) you reference the .Net Mathematica Word Library DLL (built with the above C# code) and to get the text in the Word document using this code:

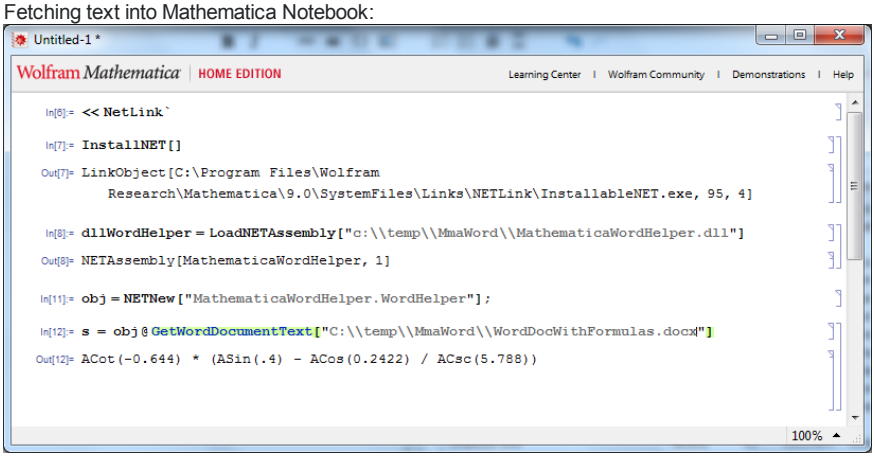
```
<< NetLink`
InstallNET[]
LoadNETAssembly["c:\\temp\\MmaWord\\MathematicaWordHelper.dll"]
obj = NETNew["MathematicaWordHelper.WordHelper"];

wordsInDocument = obj@GetWordDocumentText["C:\\temp\\MmaWord\\WordDocWithFormulas.docx"]
```

## Result

Formula in Word:

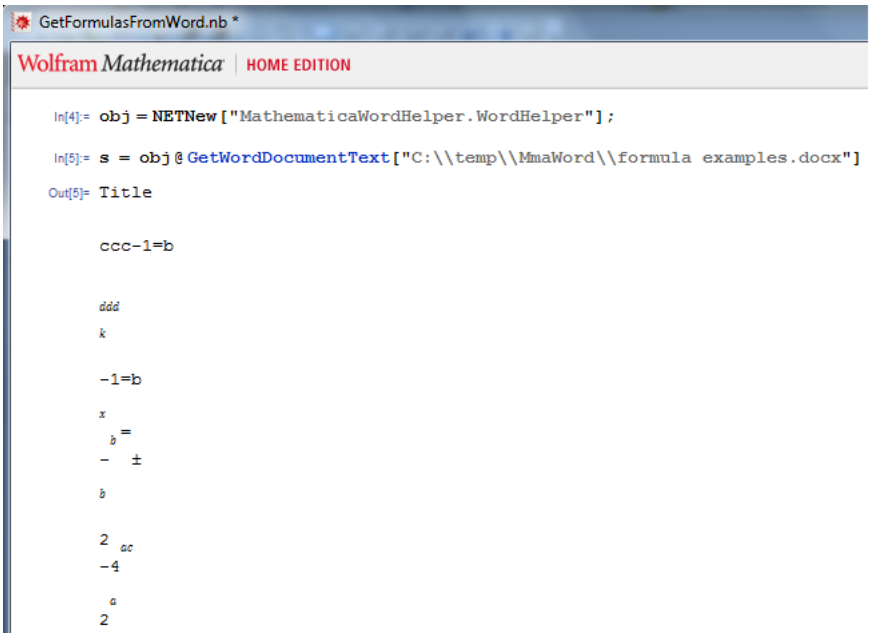



Fetching text into Mathematica Notebook:




Refer to the guide for more help:
http://reference.wolfram.com/language/NETLink/tutorial/Overview.html
http://reference.wolfram.com/language/NETLink/tutorial/CallingNETFromTheWolframLanguage.html#23489

## But importing the formulas (as words not XML Math ML) from Word looks wack!



OK, I see the problem you are having with equations involving two-dimensional layout structures, **BUT** hang on, whats this?!! Our friendly fellow Mathematica community members are suggesting *MathML?!*

*ps this is a well known issue with Microsoft and Wolfram, for example if you copy a Mathematica line into Word or Outlook it comes out in this wierd format. And as we see above, fetching data from MS Word into Mathematica is in an even more crazy format!*

## The MathML XML Method

I added `GetWordDocumentAsMathML` and `GetWordDocumentAsMathMLFormulas` methods and included the referenced DLLs and the .Net Project in the download:
http://JeremyThompson.net/Rocks/Mathematica/MmaWord.zip

So now we try to get the formula from Mathematica:

```
s1 = obj@GetWordDocumentAsMathML[
   "C:\\temp\\MmaWord\\FormulaExamples.docx", "15"]

ImportString[
  StringReplace[
   s1, {"mml:" -> "", Except[StartOfString, "<"] -> "\n<"}],
  "MathML"] // ToExpression[#1, StandardForm, HoldForm] &
```

But oh no, it combines all the formula's:

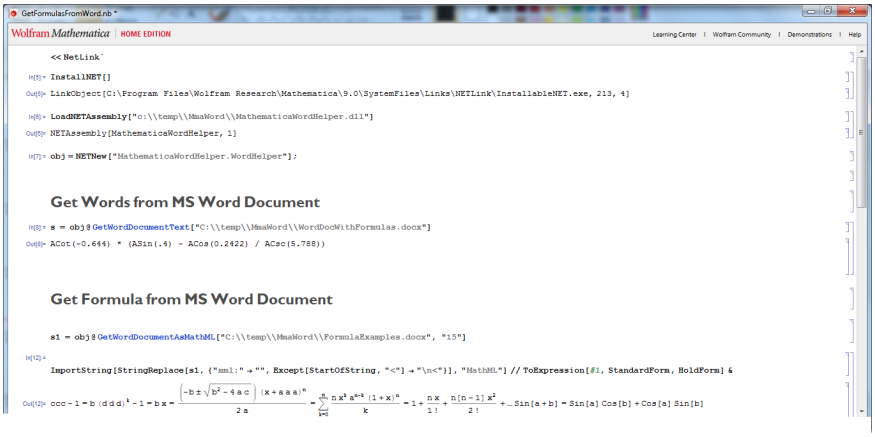
```
s1 = obj@GetWordDocumentAsMathML["C:\\temp\\MmaWord\\FormulaExamples.docx", "15"]

ImportString[StringReplace[s1, {"mml:" → "", Except[StartOfString, "<"] → "\n<"}], "MathML"] // ToExpression[#1, StandardForm, HoldForm] &

ccc - 1 = b (d d d)^k - 1 = b x = (-b ± √(b^2 - 4 a c)) (x + a a a)^n / 2 a = Σ_{k=0}^{n} n x^k a^{n-k} (1 + x)^n / k = 1 + n x / 1! + n[n - 1] x^2 / 2! + … Sin[a + b] = Sin[a] Cos[b] + Cos[a] Sin[b]
```

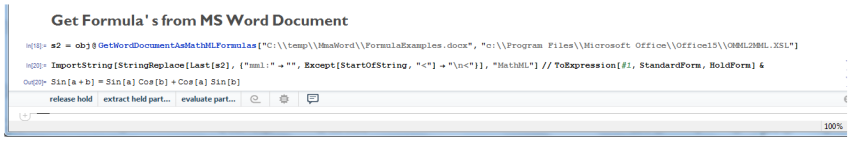
In this case we need to call the third .Net DLL method `GetWordDocumentAsMathMLFormulas` from Mathematica (this time I am using the overload which allows me to specify the full path of the XSL file), both methods have these overloads as per the C# code:

```
s2 = obj@GetWordDocumentAsMathMLFormulas[
   "C:\\temp\\MmaWord\\FormulaExamples.docx",
   "c:\\Program Files\\Microsoft Office\\Office15\\OMML2MML.XSL"]

ImportString[
  StringReplace[
   Last[s2], {"mml:" -> "", Except[StartOfString, "<"] -> "\n<"}],
  "MathML"] // ToExpression[#1, StandardForm, HoldForm] &
```

**Pay attention to "Last[s2]" in the above Mathematica query**

In summary we now have three methods to extract data from Word. 1) Get the Words, 2) Get the Equations altogether, 3) Get the Equations as a string array.

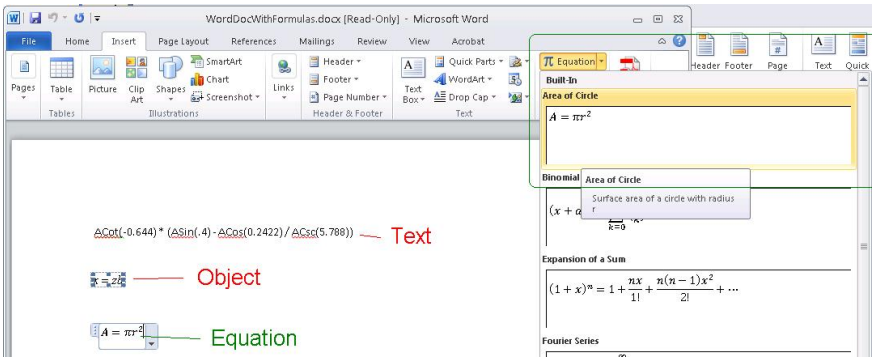



## Why dont I get any MathML returned?

If only the header MathML XML is returned, it is because there are no equations in the document:

```
<mml:math xmlns:mml="w3.org/1998/Math/MathML";
xmlns:m="schemas.openxmlformats.org/officeDocument/2006/math"; />
```




edited Nov 17 '14 at 11:06

answered Nov 14 '14 at 13:21

WolframFan

**772** 7 20

Hello @WolframFan, Thank so much for your generous help!!! It works very well for one-dimensional string formular, but it does't work with tow-dimensional layout structures(math formula as BuildingBlockEntries type in Word2013). I´m desperately trying to solve this prolem..... anybody? Thanks a lot again! – Xiang Li Nov 15 '14 at 8:47

Hi, I would be happy to help the only problem is I cant see your MS Word document. Can you upload it to temp-share or something like that for me to view it? When I try your url/link this is the result (just XML not a downloadable word document): This XML file does not appear to have any style information associated with it. The document tree is shown below. ExpiredToken The provided token has expired.() /download.weipan.cn/2922479/18673fc198712703919a59833bb1a36dfecabaaf 00907f51-1411-1420-3915-782bcb71fffa – WolframFan Nov 15 '14 at 9:59

Hi, @WolframFan . I have just stored example docx into Dropbox. I hope, It can download correctly. Thanks! – Xiang Li Nov 15 '14 at 10:21

1 It looks like the WordprocessingDocument is this from the Open XML SDK. I had a quick look at it too, but I'm not at all familiar with .NET and it seemed too much like hard work for a Sunday distraction :-) – Simon Woods Nov 16 '14 at 12:31

1 @WolframFan I really really appreciate your help! – Xiang Li Dec 20 '14 at 3:32