XML to HTML Converter

June 14, 2021 4:20 PM

Problem Description:
Write a simple C++ console program that takes cd_catalog.xml (attached) as input and converts it to an HTML file.

Requirements:

- **Document and explain your design choices. This can be done with in-code comments, dedicated documentation files (i.e. "README"), or both.**
- It should be a single executable that takes the file name as input
- The output HTML should be formatted as a "table" element
- Sorting is not important for this challenge

In the input XML file the data is given as below

C:\Programming\cd_catalog.xml - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins

cd_catalog.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CATALOG>
    <CD>
        <TITLE>Mad Dogs and Englishmen</TITLE>
        <ARTIST>Joe Cocker</ARTIST>
        <COMPANY>A&amp;amp;M</COMPANY>
        <COUNTRY>USA</COUNTRY>
        <PRICE>9.80</PRICE>
        <YEAR>1970</YEAR>
    </CD>
    <CD>
        <TITLE>Empire Burlesque</TITLE>
        <ARTIST>Bob Dylan</ARTIST>
        <COUNTRY>USA</COUNTRY>
        <COMPANY>Columbia</COMPANY>
        <PRICE>10.90</PRICE>
        <YEAR>1985</YEAR>
    </CD>
```

This information is pertaining to CDs. The information need to be converted into HTML Table format as shown below.

C:\Users\fakig\source\repos\XMLtoHTML\x64\Release\Output.html - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

cd_catalog.xml | temp.xhtml | cdcatalog.xsl | cd_catalog.xml | tinyxml2.h | cd_catalog2

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
</style>

<body>
<table>
<tr>
<th>TITLE</th>
<th>ARTIST</th>
<th>COMPANY</th>
<th>COUNTRY</th>
<th>PRICE</th>
<th>YEAR</th>
</tr>
<tr>
<td>Mad Dogs and Englishmen</td>
<td>Joe Cocker</td>
<td>A&M</td>
<td>USA</td>
<td>9.80</td>
<td>1970</td>
</tr>
```

When opened in browser the table will look like below

TITLE	ARTIST	COMPANY	COUNTRY	PRICE	YEAR
Mad Dogs and Englishmen	Joe Cocker	A&M	USA	9.80	1970
Empire Burlesque	Bob Dylan	Columbia	USA	10.90	1985
Hide your heart	Bonnie Tyler	CBS Records	UK	9.90	1988
Greatest Hits	Dolly Parton	RCA	USA	9.90	1982
Still got the blues	Gary Moore	Virgin records	UK	10.20	1990
Eros	Eros Ramazzotti	BMG	EU	9.90	1997
One night only	Bee Gees	Polydor	UK	10.90	1998
Sylvias Mother	Dr.Hook	CBS	UK	8.10	1973
Maggie May	Rod Stewart	Pickwick	UK	8.50	1990
Romanza	Andrea Bocelli	Polydor	EU	10.80	1996
When a man loves a woman	Percy Sledge	Atlantic	USA	8.70	1987
Black angel	Savage Rose	Mega	EU	10.90	1995
1999 Grammy Nominees	Many	Grammy	USA	10.20	1999
For the good times	Kenny Rogers	Mucik Master	UK	8.70	1995
Tupelo Honey	Van Morrison	Polydor	UK	8.20	1971
Soulsville	Jorn Hoel	WEA	Norway	7.90	1996
The very best of	Cat Stevens	Island	UK	8.90	1990
Stop	Sam Brown	A and M	UK	8.90	1988
Bridge of Spies	T'Pau	Siren	UK	7.90	1987
Private Dancer	Tina Turner	Capitol	UK	8.90	1983
Midt om natten	Kim Larsen	Medley	EU	7.80	1983
Pavarotti Gala Concert	Luciano Pavarotti	DECCA	UK	9.90	1991
The dock of the bay	Otis Redding	Atlantic	USA	7.90	1987
Picture book	Simply Red	Elektra	EU	7.20	1985
Red	The Communards	London	UK	7.80	1987
Unchain my heart	Joe Cocker	EMI	USA	8.20	1987

Based on Requirement , C++ console application need to be developed to convert the XML data into HTML table data

After doing exhaustive search few promising libraries found that can do the conversion with less code changes

Library Search:

MSXML:
It has a good API that can directly convert XML into HTML but it has dependence on Microsoft libraries. Also MSXML require additional XSLT (template) document . After initial investigation(tried with some prototype code) it was found that using MSXML will not help much as code is quickly getting ugly

Rapidxml:
It is a light weight application but it looks like learning curve to use this library is high. Due to the current time constrains this is not recommended

Tinyxml:
i)This library has very good example and it contains only couple of(source) files. We don't need to link the dlls into the project We can just simply put the Library source code with the project source code
Ii)It is also very easy to parse through the document with Tinyxml library

Pugixml:
Similar to RapidXml this library interface doesn't look more interesting than Tiny XML

Out of above libraries TInyXml library has very good advantages compared to any other libraries based on constrain and application

Analysis:

I have investigated few ways to implement the conversion from XML to HTML

Option 1: Conversion using XSLT:

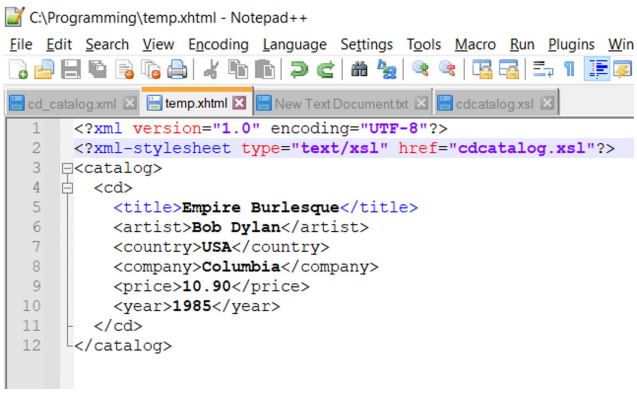
This conversion can be easily achieved by using XSLT - Transformation function as mentioned in this tutorial

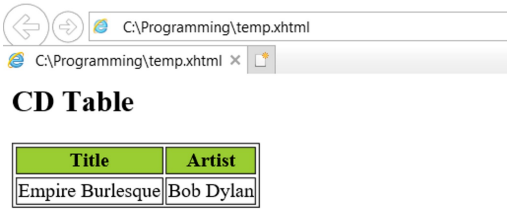
https://www.w3schools.com/xml/xsl_transformation.asp

Following XML Style sheet template can be used to convert XML to XHTML

```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
 <html>
 <body>
 <h2>My CD Collection</h2>
 <table border="1">
  <tr bgcolor="#9acd32">
   <th>Title</th>
   <th>Artist</th>
  </tr>
  <xsl:for-each select="catalog/cd">
  <tr>
   <td><xsl:value-of select="title"/></td>
   <td><xsl:value-of select="artist"/></td>
  </tr>
  </xsl:for-each>
 </table>
 </body>
 </html>
</xsl:template>

</xsl:stylesheet>
```

If reference is used in the original XML file, the browser will automatically convert into HTML





There is a main disadvantage in this option as the final output is of the form of XHTML. But actual requirement is to convert XML into HTML Table format.

Advantages:
Simple straight forward solution using online tools

Disadvantage:
No open source libraries available for conversion between XHTML to HTML
Requires definition of XLMS Template if input data changes it requires new template
Not a straight forward or clean solution
Doesn't comply with the requirements

Performance:
Not ideal for converting large data
Doesn't support multi threading

Scalability:
Not scalable

Option 2: Conversion using MSXML:

Microsoft XML library provides various functionalities including XML Conversion. But the solution is not straight forward. Also code is heavily platform dependent on Microsoft libraries.

Advantages:
Library is readily available
Already tested code

Disadvantage:
Implementation is not tidy
Not a straight forward or clean solution
Requires definition of XLMS Template. if input data changes then it require a new template

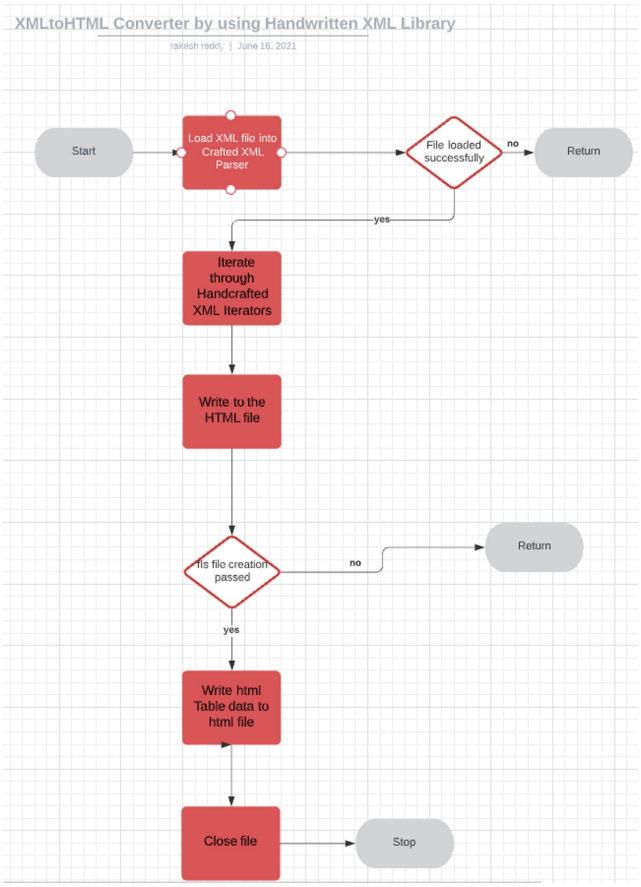
Performance:
Depends on many Microsoft libraries
Thread safety can't be guaranteed

Scalability:
Can be scalable

Option 3: Direct conversion using own parser(Starting from wheel approach)

We can write our own parser than can read(XML)/write(HTML) file using standard 'fstream' library. This option will require lot of effort as we have to do the XML Schema validations and write our own Tree iterators(for XML parsing) e.t.c.. Also requires lot of testing

If this implemented, The Flow chart would be as below



As explained in the flow chart , We first need to load the XML file into C++ application using standard file operations. In the next step we will parse the XML document with out own iterators to write HTML Table contents directly into html file

Advantage:
We have better control about the Library source code
We can add in very good thread support

Disadvantages:
We have to stick to only one format of the XML file. We need a generic application that can load all types of xml files
Time consuming
More testing needed as we are the only one creating and using the library
Requires lots of effort. We are rewriting the XML library

Scalability:
Code can be made scalable but it has to be in incremental way

Performance:
We may have an edge in performance as we are writing our own library

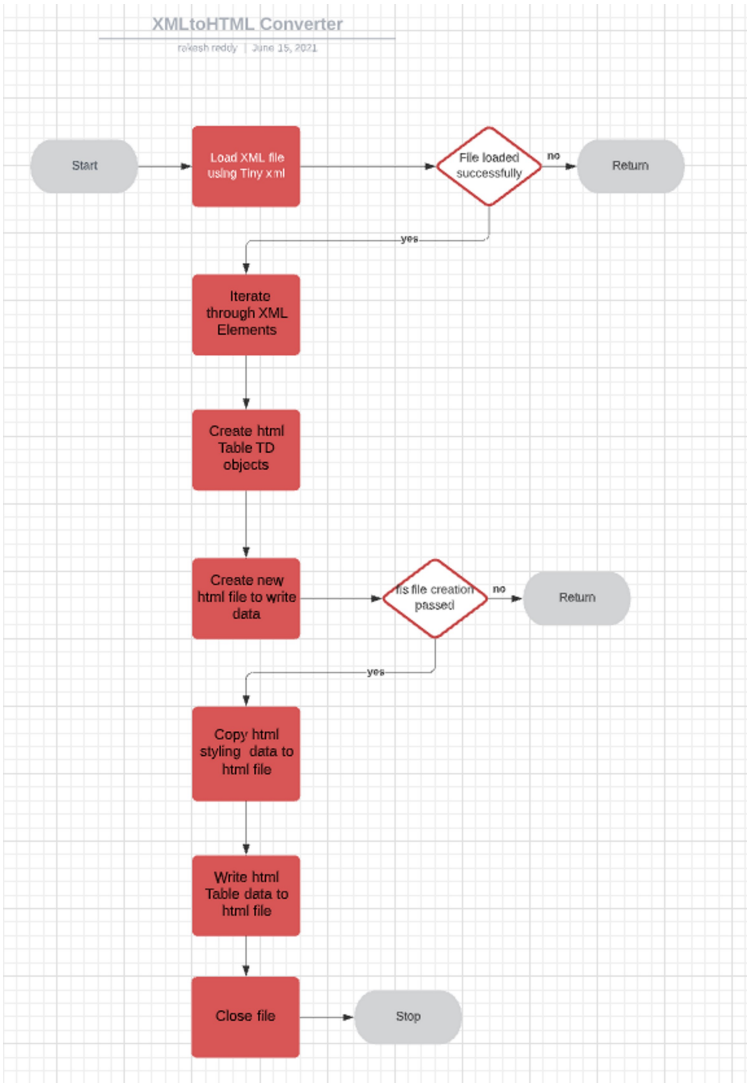
This is the design option that we are choosing to implement

Option 4: Conversion using TinyXML2 library - The Current implementation

Tinyxml2 library is extracted from below. There are very good tutorials on how to use the library to quickly parse through the XML document and access to the XML Elements. Since the messier part has already taken cared by the library we can focus on the application part of the code
http://www.grinninglizard.com/tinyxml2/

Algorithm flow is clearly explained in the below diagram. XML file is read into our C++ application using a TinyXMl Wrapper class(XMLProcessor). After reading the file we can use the tinyxml Elements to iterate through the XML Elements. While

iterating through, We will create C++ html data objects instead of directly writing into output file. Then these data objects will be later serial outed into html file with some configurable styling. This has very good advantage in terms of scalability , improving generic abilities and testing capabilities as well. On top of that instead of directly doing all out executions in our main method we have opportunity to work out few classes that will help on scalability and performance. Further details about the classes will be added after this section



To further explain the generic capabilities of the algorithm,

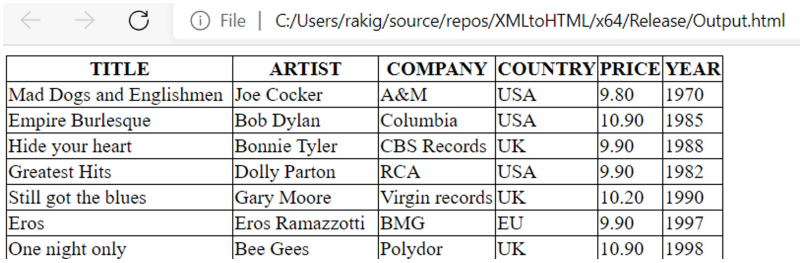
1) If the input XML file changes algorithm will still work because Algorithm is independent of XML data
2) Since we are creating intermediary HTML C++ objects we can fix the data ordering problems as given below

```
<CATALOG>
    <CD>
        <TITLE>Mad Dogs and Englishmen</TITLE>
        <ARTIST>Joe Cocker</ARTIST>
        <COMPANY>A&amp;amp;M</COMPANY>
        <COUNTRY>USA</COUNTRY>
        <PRICE>9.80</PRICE>
        <YEAR>1970</YEAR>
    </CD>
    <CD>
        <TITLE>Empire Burlesque</TITLE>
        <ARTIST>Bob Dylan</ARTIST>
        <COUNTRY>USA</COUNTRY>
        <COMPANY>Columbia</COMPANY>
        <PRICE>10.90</PRICE>
        <YEAR>1985</YEAR>
    </CD>
    <CD>
```

In the given input the first entry of CD data the third entry is for the <COMPANY> tag where as in the second CD information the third entry is <COUNTRY>. When we directly flush out to HTML we will treat the third row information as <COMPANY> for both entries resulting incorrect display in the final output html file. We can fix this problem as we could store the information in the C++ object as a map.

Another feature of this algorithm is that it supports the configurable HTML Table styling. The application take a styling input as a txt file if no styling input is provided it will use the default styling

Default styling is below



TITLE	ARTIST	COMPANY	COUNTRY	PRICE	YEAR
Mad Dogs and Englishmen	Joe Cocker	A&M	USA	9.80	1970
Empire Burlesque	Bob Dylan	Columbia	USA	10.90	1985
Hide your heart	Bonnie Tyler	CBS Records	UK	9.90	1988
Greatest Hits	Dolly Parton	RCA	USA	9.90	1982
Still got the blues	Gary Moore	Virgin records	UK	10.20	1990
Eros	Eros Ramazzotti	BMG	EU	9.90	1997
One night only	Bee Gees	Polydor	UK	10.90	1998

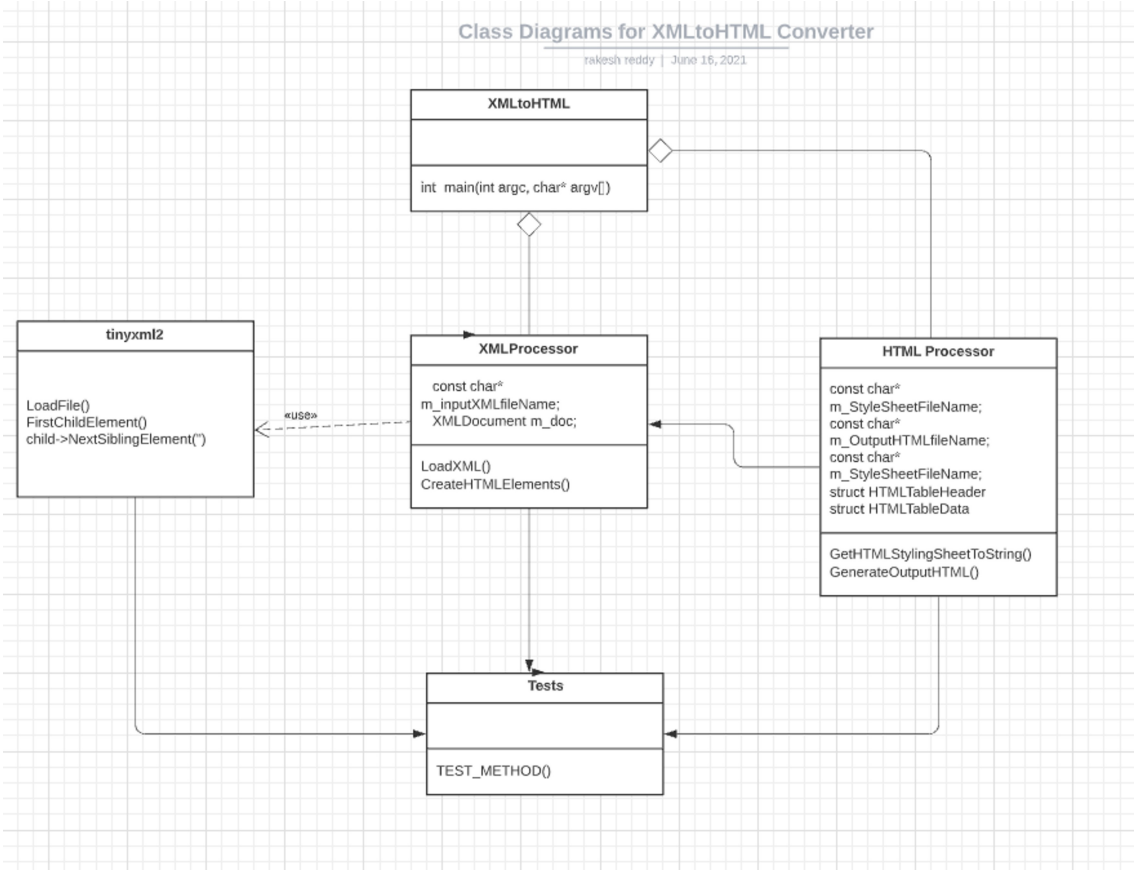
With Styling input

TITLE	ARTIST	COMPANY	COUNTRY	PRICE	YEAR
Mad Dogs and Englishmen	Joe Cocker	A&M	USA	9.80	1970
Empire Burlesque	Bob Dylan	Columbia	USA	10.90	1985
Hide your heart	Bonnie Tyler	CBS Records	UK	9.90	1988
Greatest Hits	Dolly Parton	RCA	USA	9.90	1982

Features of this Design:

Configurability in terms of HMI - Styling Text support
Code reusability- Same classes are used for testing and implementation
Future compatibility- Current classes can be easily extended for future releases
Support for scaling- Since we are creating C++ objects for each XML nodes the application can handle large amount of data
Support for multi threading - We can easily add multi threading support by simply diving the work between the threads
Input Independent- Any XML to HTML converter
Added support for Test framework - Classes can be easily extended to be tested with unit tests
Better code coverage- Code coverage is improved since code is modular
Using very Light weight XML parsing library- We are using Tinyxml2(only two files and not platform independent)
Simple design- very easy design with the total of 5 classes
Build is supported by both with Makefile and with VisualStudio solution file - easy for CI and CD
Tests can be easily run on cloud with Jenkins jobs or in pipelines as batch script is created to run tests automatically

CLASS DIAGRAMS:



Class responsibilities:

XMLtoHTML:
XMLtoHTML is the entrant class where main method is implemented. This class is responsible to create XMLProcessor and HTMLProcessor objects . This class is also responsible to maintain the flow of the data

XMLProcessor:
This class is responsible to Load and parse XML file. This class is also responsible to create HTML data objects after iterating through the XML nodes

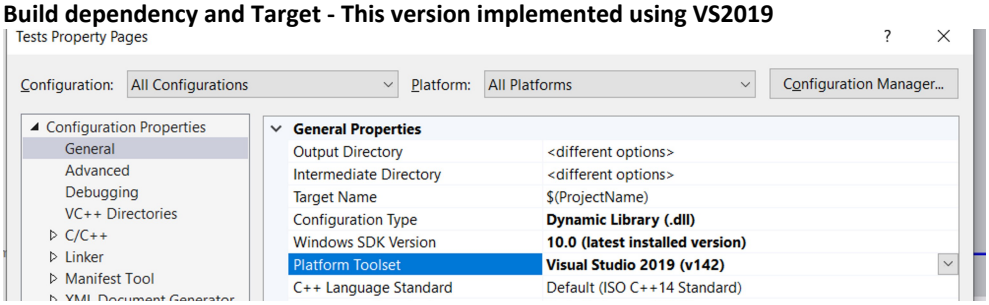
HTMLProcessor:
This class is responsible to generate and export HTML content. This is also responsible to add additional styling graphics based on configuration

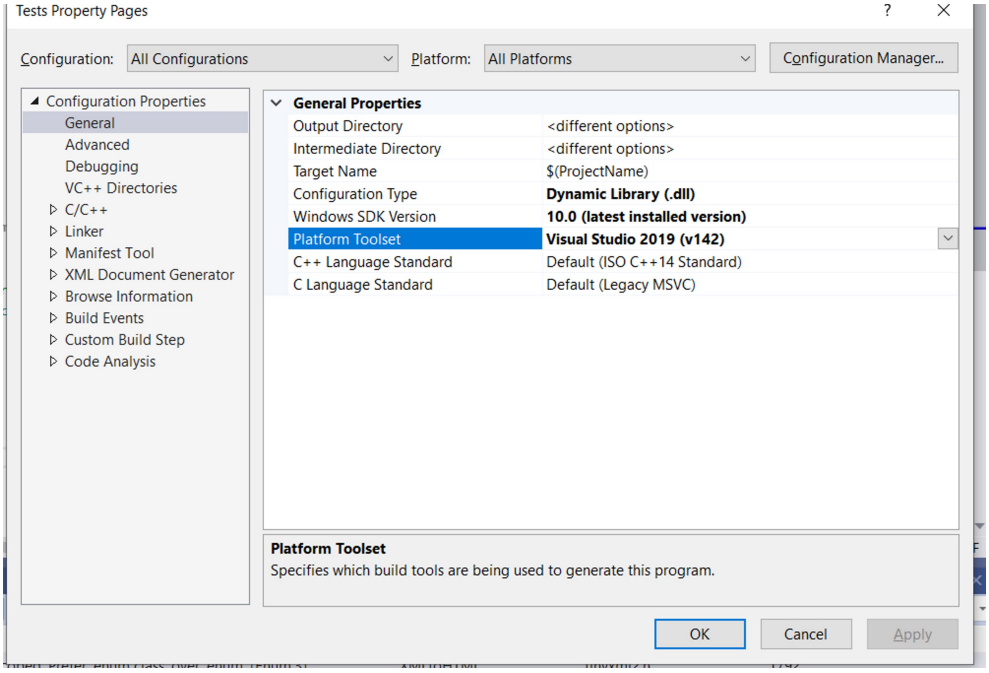
Tinyxml2:
This is not a new class we are just refencing here

Tests:
This class is responsible to connect to the necessary elements to build and run tests

Implementation details:

- **Build dependency and Target - This version implemented using VS2019**

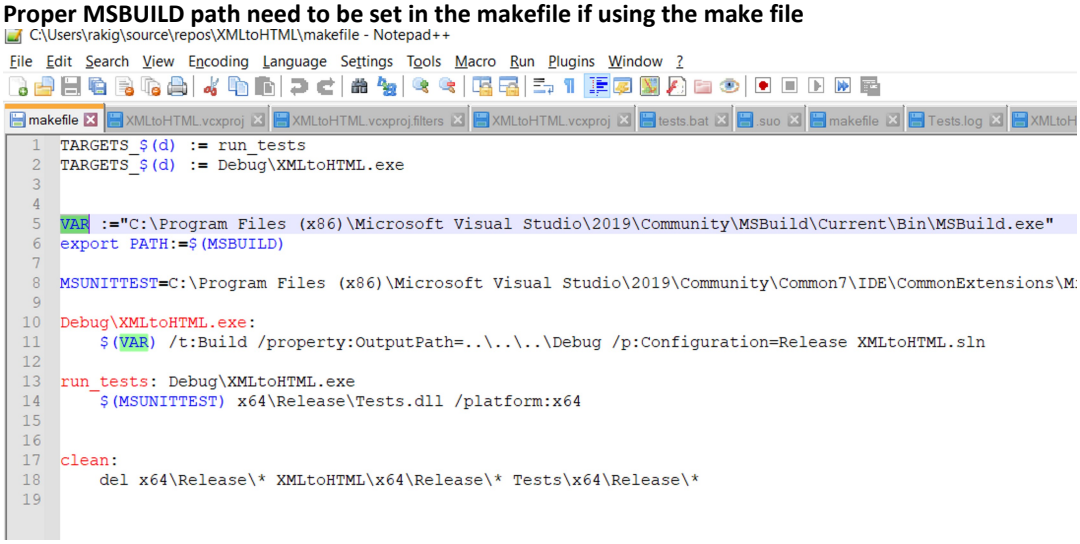




If try to build on different versions there might be some issues

- **This source code is implemented using VS2019 with standard C++ 2014 standard**
- **Makefile depends on MSBUILD**

Proper MSBUILD path need to be set in the makefile if using the make file

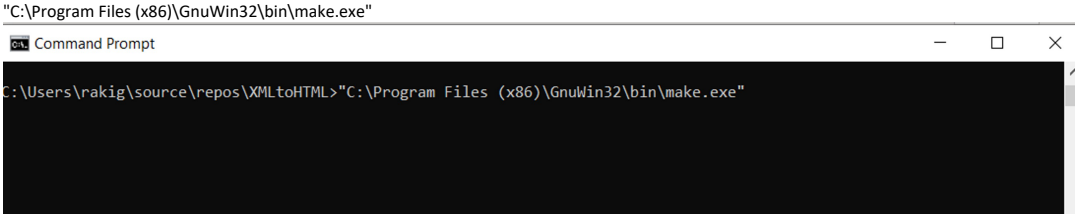


Build can also be run by loading Visual Studio Solution file and using the build command from the menu

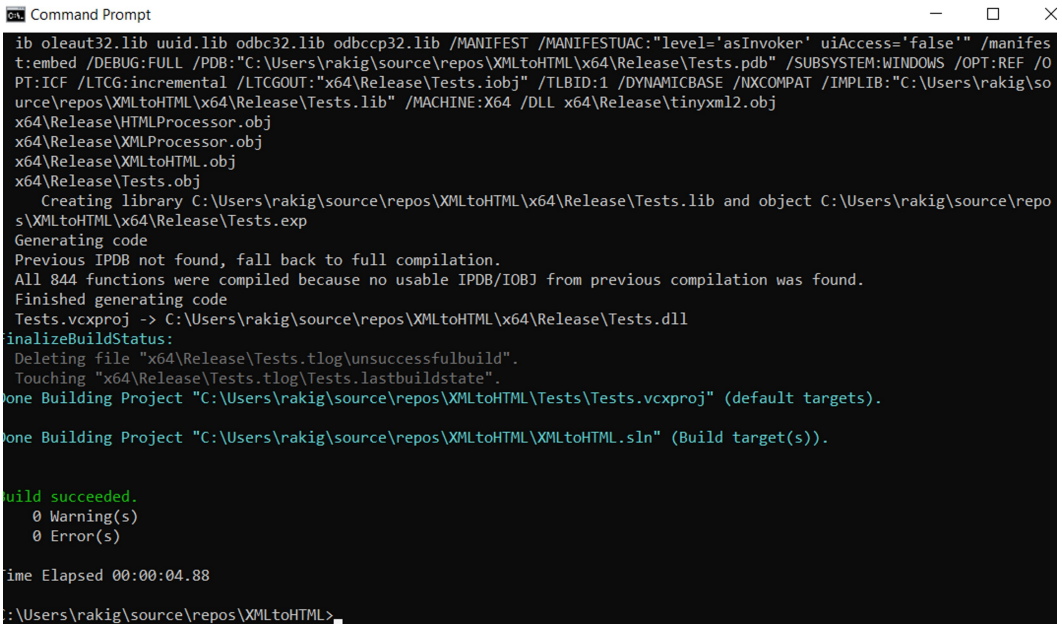
How to Build:

Get the source from Repo and from the root run

"C:\Program Files (x86)\GnuWin32\bin\make.exe"

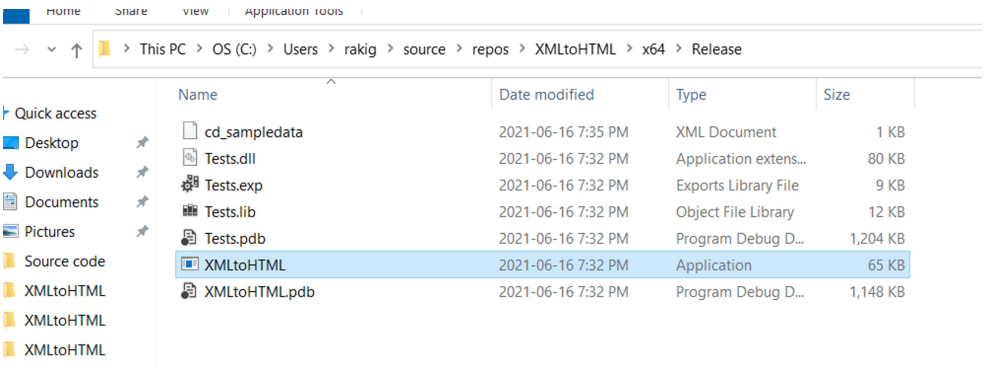


Note that the build hasn't been tested with GCC compiler due to time constrains

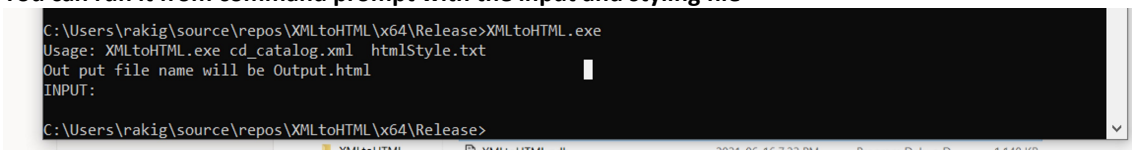


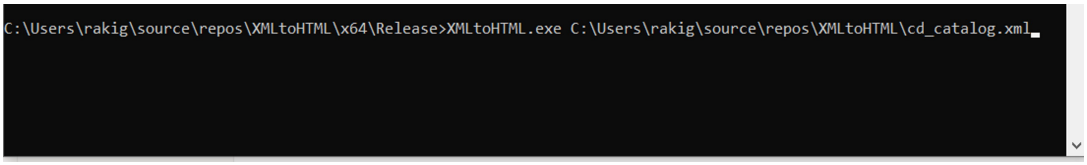
How to test Manually:

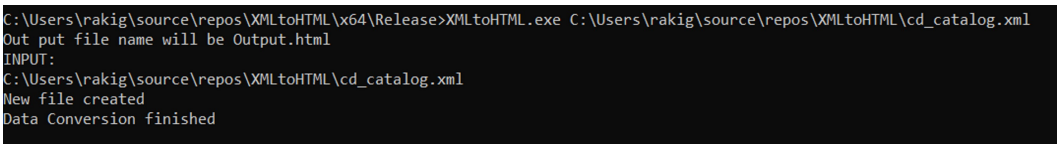
The .exe is generated in the Release folder

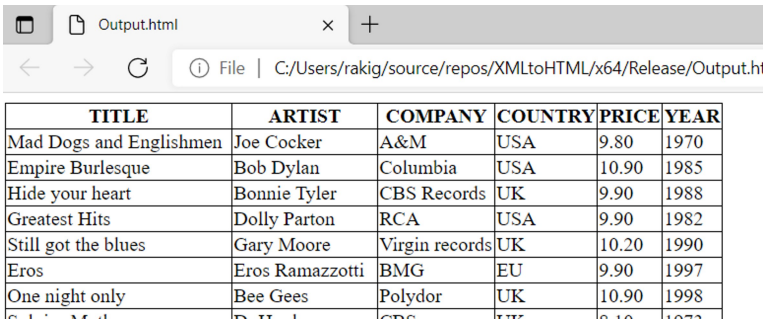


You can run it from command prompt with the input and styling file









TITLE	ARTIST	COMPANY	COUNTRY	PRICE	YEAR
Mad Dogs and Englishmen	Joe Cocker	A&M	USA	9.80	1970
Empire Burlesque	Bob Dylan	Columbia	USA	10.90	1985
Hide your heart	Bonnie Tyler	CBS Records	UK	9.90	1988
Greatest Hits	Dolly Parton	RCA	USA	9.90	1982
Still got the blues	Gary Moore	Virgin records	UK	10.20	1990
Eros	Eros Ramazzotti	BMG	EU	9.90	1997
One night only	Bee Gees	Polydor	UK	10.90	1998

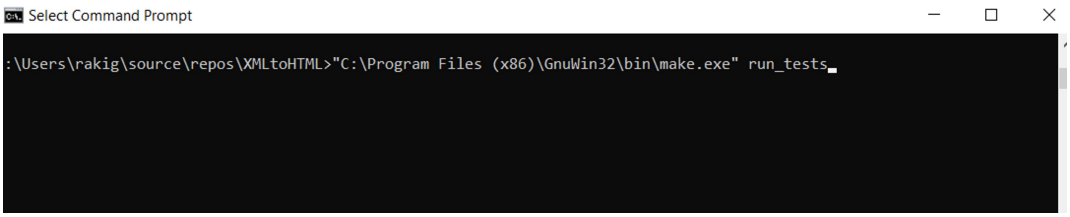
With the Styling file

```
C:\Users\rakig\source\repos\XMLtoHTML\x64\Release>XMLtoHTML.exe C:\Users\rakig\source\repos\XMLtoHTML\cd_catalog.xml C:\Users\rakig\source\repos\XMLtoHTML\htmlStyle.txt
Out put file name will be Output.html
INPUT:
C:\Users\rakig\source\repos\XMLtoHTML\cd_catalog.xml
New file created
249 characters read...
Data Conversion finished
```

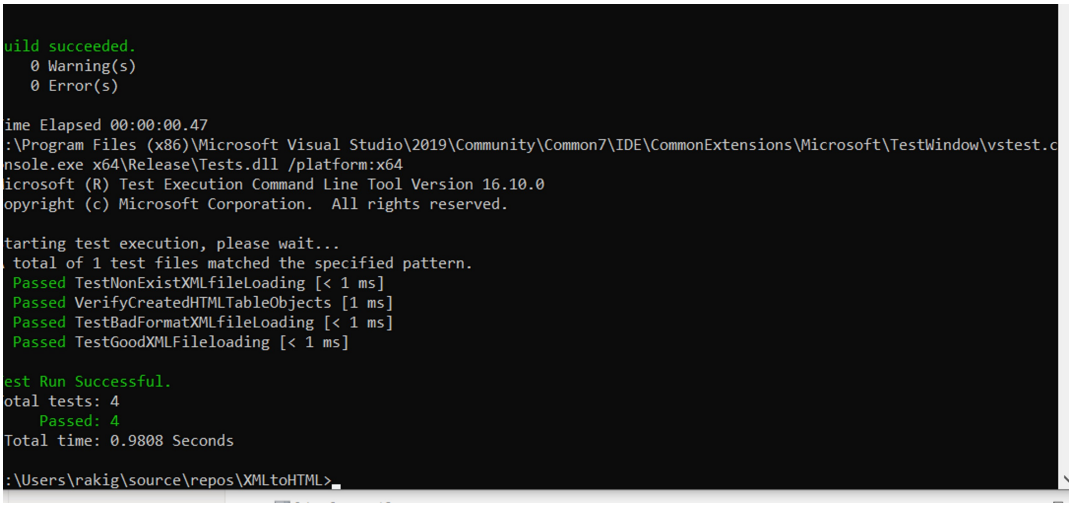
TITLE	ARTIST	COMPANY	COUNTRY	PRICE	YEAR
Mad Dogs and Englishmen	Joe Cocker	A&M	USA	9.80	1970
Empire Burlesque	Bob Dylan	Columbia	USA	10.90	1985
Hide your heart	Bonnie Tyler	CBS Records	UK	9.90	1988
Greatest Hits	Dolly Parton	RCA	USA	9.90	1982
Still got the blues	Gary Moore	Virgin records	UK	10.20	1990

How to Run Automated Tests:

Tests can be run with the Target "run_tests"



All tests are passing



With the current tests we have There is a good converge for the coding. This can be improved further

Future Improvements :

1)Multithreading support can be added to LoadXML() method to create multiple HTML data objects for faster processing((large files)
2))Multithreading support can be added to GenerateOutputHTML() method to create large HTML files faster
2)Code is platform independent but it was never tested with GCC compiler. It can be made easily portable
3)More tests can be added at the GUI level. Currently we are not testing the graphics output of the html browser
4)Application should work for different input XML files however it was not completely tested
5)When static analyzer ran on the code it was observed that there are few warnings in the tinyxml2 library source code. These need to be investigated. There are no Errors or Warnings reported against the new code

```
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.h(1164): warning C26812: The enum type 'tinyxml2::XMLError' is unscoped. Prefer 'enum class'
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.h(1677): warning C26812: The enum type 'tinyxml2::XMLElement::ElementClosingType' is unscope
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.h(1792): warning C26812: The enum type 'tinyxml2::Whitespace' is unscoped. Prefer 'enum clas
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.cpp(1211): warning C26812: The enum type 'tinyxml2::StrPair::Mode' is unscoped. Prefer 'enum
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.cpp(2496): warning C33010: Unchecked lower bound for enum errorID used as index..
\Users\rakig\source\repos\XMLtoHTML\XMLtoHTML\External Dependencies\tinyxml2.h(209): warning C26495: Variable 'tinyxml2::DynArray<char,20>::_pool' is uninitialized. Alwa
```