



The following example shows how to parse XML file using Symbian OS C++ class, CParser. CParser is basically a SAX (Simple API for XML)-based XML parser.

It uses an active object to read the XML file chunk by chunk (see `CXmlHandler::StartParsingWithAoL()` method). On each chunk, it passes the buffer to the XML parser. When the XML parser finds an element, it calls the respective callback functions, for example `CXmlHandler::OnStartElementL()` or `CXmlHandler::OnEndElementL()`.

To use CParser class, the `XmlFramework.lib` has to be included in the `.mmp` file. For more information about CParser, please visit some links at the end of this page.

Contents

- [1](#)
[XmlHandler.h](#)
- [2](#)
[XmlHandler.cpp](#)
- [3](#) [download](#)
[example](#)
- [4](#) [See Also](#)

XmlHandler.h

```
#ifndef __XMLHANDLER_H__
#define __XMLHANDLER_H__

// INCLUDE FILES
#include <e32base.h>
#include <f32file.h> //Link against efsrv.lib
#include <xml\contenthandler.h> // for MContentHandler
#include <xml\parser.h> // for CParser

// CLASS DECLARATION

using namespace Xml;

class CXmlHandler: public CActive, MContentHandler
{
public: // Constructors and destructor

    static CXmlHandler* NewL();

    static CXmlHandler* NewLC();

    virtual ~CXmlHandler();

public: // Public methods

    void StartParsingWithAoL( const TDesC& aFileName );
```

How_to_parse_XML_file_using_CParser_class

```
private: // Constructors

    CXmlHandler();

    void ConstructL();

private: // from CActive

    void DoCancel();

    void RunL();

private: // from MContentHandler

    void OnStartDocumentL( const RDocumentParameters &aDocParam,
        TInt aErrorCode );

    void OnEndDocumentL( TInt aErrorCode );

    void OnStartElementL( const RTagInfo &aElement,
        const RAttributeArray &aAttributes, TInt aErrorCode );

    void OnEndElementL( const RTagInfo &aElement, TInt aErrorCode );

    void OnContentL( const TDesC8 &aBytes, TInt aErrorCode );

    void OnStartPrefixMappingL( const RString &aPrefix, const RString &aUri,
        TInt aErrorCode );

    void OnEndPrefixMappingL( const RString &aPrefix, TInt aErrorCode );

    void OnIgnorableWhiteSpaceL( const TDesC8 &aBytes, TInt aErrorCode );

    void OnSkippedEntityL( const RString &aName, TInt aErrorCode );

    void OnProcessingInstructionL( const TDesC8 &aTarget, const TDesC8 &aData,
        TInt aErrorCode);

    void OnError( TInt aErrorCode );

    TAny *GetExtendedInterface( const TInt32 aUid );

private: // Private data

    CParser*          iParser;
    HBufC8*          iBuffer;
    RFile             iFile;

};

#endif /* __XMLHANDLER_H__ */
```

XmlHandler.cpp

```
// INCLUDE FILES
#include <coemain.h>
#include "XmlHandler.h"
```

XmlHandler.h

How_to_parse_XML_file_using_CParser_class

```
// CONSTANTS
const TInt KFileBufferSize = 1024; // buffer size for file reading
_LIT8( KXmlMimeType, "text/xml" );

// METHODS DEFINITION

CXmlHandler* CXmlHandler::NewL()
{
    CXmlHandler* self = CXmlHandler::NewLC();
    CleanupStack::Pop();
    return self;
}

CXmlHandler* CXmlHandler::NewLC()
{
    CXmlHandler* self = new ( ELeave ) CXmlHandler();
    CleanupStack::PushL( self );
    self->ConstructL();
    return self;
}

CXmlHandler::~~CXmlHandler()
{
    Cancel();
    delete iParser;
    delete iBuffer;
}

CXmlHandler::CXmlHandler()
    :CActive( EPriorityStandard )
{
    CActiveScheduler::Add( this );
}

void CXmlHandler::DoCancel()
{
    iParser->ParseEndL();
    iFile.Close();
    delete iBuffer;
    iBuffer = 0;
}

void CXmlHandler::RunL()
{
    if ( KErrNone == iStatus.Int() )
    {
        // If the buffer length is zero, it means if we have reached
        // the end of the file.
        if ( iBuffer->Length() == 0 )
        {
            iParser->ParseEndL();
            iFile.Close();
            delete iBuffer;
            iBuffer = 0;
        }

        // Otherwise, we continue reading the next chunk of the XML file.
        else
        {
            // Parse the next "part" of the XML document.
            iParser->ParseL( *iBuffer );
        }
    }
}
```

How_to_parse_XML_file_using_CParser_class

```
        // Read the next chunk of the file.
        TPtr8 bufferPtr( iBuffer->Des() );
        iFile.Read( bufferPtr, KFileBufferSize, iStatus );

        // Don't forget to call this... :)
        SetActive();
    }
}
else
{
    // Do something if error happens.
}
}

void CXmlHandler::ConstructL()
{
    iParser = CParser::NewL( KXmlMimeType, *this );
}

void CXmlHandler::StartParsingWithAoL( const TDesC& aFileName )
{
    // Remember to cancel any outstanding request first.
    if ( IsActive() )
    {
        Cancel();
    }

    User::LeaveIfError( iFile.Open( CCoeEnv::Static()->FsSession(), aFileName,
        EFileRead ) );

    // Create a buffer to store the file content.
    // Note that this method uses active object to read the file.
    // So we have to call SetActive() at the end. Then we call CParser::ParseL()
    // in RunL() method.
    delete iBuffer;
    iBuffer = 0;
    iBuffer = HBufC8::NewL( KFileBufferSize );
    TPtr8 bufferPtr( iBuffer->Des() );
    iFile.Read( bufferPtr, KFileBufferSize, iStatus );
    SetActive();

    // Tell the parser that we are about to parse a XML document.
    iParser->ParseBeginL();
}

void CXmlHandler::OnStartDocumentL( const RDocumentParameters& /*aDocParam*/,
    TInt aErrorCode )
{
    if ( KErrNone == aErrorCode )
    {
        // Do something here when the parser at the start of the document.
    }
    else
    {
        // Do something if error happens.
    }
}

void CXmlHandler::OnEndDocumentL( TInt aErrorCode )
{
    if ( KErrNone == aErrorCode )
```

How_to_parse_XML_file_using_CParser_class

```
{
    // Do something here when the parser reaches the end of the document.
}

void CXmlHandler::OnStartElementL( const RTagInfo& aElement,
    const RAttributeArray& /*aAttributes*/, TInt aErrorCode )
{
    if ( KErrNone == aErrorCode )
    {
        // Found start of an element, for example: "<tag>"
        // The name of the element is stored in aElement.LocalName().Desc().

        // Do something with the start of an element.
    }
    else
    {
        // Do something if error happens.
    }
}

void CXmlHandler::OnEndElementL( const RTagInfo &aElement, TInt aErrorCode )
{
    if ( KErrNone == aErrorCode )
    {
        // Found the end of an element, for example: "</tag>"
        // The name of the element is stored in aElement.LocalName().Desc().

        // Do something with the end of an element.
    }
    else
    {
        // Do something if error happens.
    }
}

void CXmlHandler::OnContentL( const TDesC8 &aBytes, TInt aErrorCode )
{
    if ( KErrNone == aErrorCode )
    {
        // aBytes stored the value of the parsed contents.
    }
    else
    {
        // Display error messages here.
    }
}

void CXmlHandler::OnStartPrefixMappingL( const RString& /*aPrefix*/,
    const RString& /*aUri*/, TInt aErrorCode )
{
}

void CXmlHandler::OnEndPrefixMappingL( const RString& /*aPrefix*/,
    TInt aErrorCode )
{
}

void CXmlHandler::OnIgnorableWhiteSpaceL( const TDesC8& /*aBytes*/,
    TInt aErrorCode )
{
}
```

How_to_parse_XML_file_using_CParser_class

```
void CXmlHandler::OnSkippedEntityL( const RString& /*aName*/,
    TInt aErrorCode )
{
}

void CXmlHandler::OnProcessingInstructionL( const TDesC8& /*aTarget*/,
    const TDesC8& /*aData*/, TInt aErrorCode )
{
}

void CXmlHandler::OnError( TInt aErrorCode )
{
    // Do something if error happens.
}

TAny* CXmlHandler::GetExtendedInterface( const TInt32 /*aUid*/ )
{
    return 0;
}
```

download example

- [myxmlparser.zip](#)

See Also

- [Use the XML Parser in Symbian OS 9](#)
- [Example Code of Using XML Parser in Symbian OS](#)