

```

using System;
using System.Xml;
using System.Xml.Schema;
using System.Windows.Forms;

namespace SB54_CSAI.TrackObjects
{
    #region XMLValidator CLASS
    /// <summary>
    /// Provides a mechanism for validating a XML (*.XML) against a XML schema document (*.
    XSD)
    /// This class will be used to validate <see cref="SB54_CSAI.MediaPlayerControl.
    uctMediaPlayer">Media Player tracklists </see>
    /// and <see cref="SB54_CSAI.ClientTrackList.uctClientTrackList">Client track lists </
    see> that the user may choose to open.
    /// If the XML file does not match the schema, the XML file will not
    /// be parsed
    /// </summary>
    public class XMLValidator
    {
        #region Instance Fields
        //Instance Fields
        private string XMLFileName ;
        private string SchemaFileName ;
        private XmlSchemaCollection oXmlSchemaCollection ;
        private bool IsValid=true ;
        #endregion
        #region Public Constructor
        /// <summary>
        /// Simply constructs a new XMLValidator object using the parameters
        /// provided
        /// </summary>
        /// <param name="XMLFileName">The Path to the XML file to validate</param>
        /// <param name="SchemaFileName">The Path to the XML Schema file to use to validate
        </param>
        public XMLValidator (string XMLFileName, string SchemaFileName)
        {
            this.XMLFileName = @XMLFileName;
            this.SchemaFileName = @SchemaFileName;
            oXmlSchemaCollection = new XmlSchemaCollection ();
            //adding the schema file to the newly created schema collection
            oXmlSchemaCollection.Add (null, SchemaFileName);
        }
        #endregion
        #region Public Methods
        /// <summary>
        /// Validates the XML file passed into the constructor against the XML schema file
        passed
        /// into the constructor
        /// </summary>
        /// <returns>true if the XML file passed into the constructor is valid when compared
        against
        /// the XML schema file passed into the constructor</returns>
        public bool ValidateXMLFile()
        {
            XmlTextReader oXmlTextReader =null;
            XmlValidatingReader oXmlValidatingReader=null ;

            try
            {
                //creating a text reader for the XML file already picked by the
                //overloaded constructor
                oXmlTextReader = new XmlTextReader(XMLFileName);
                //creating a validating reader for that oXmlTextReader just created
                oXmlValidatingReader = new XmlValidatingReader (oXmlTextReader);
                //For validation we are adding the schema collection in
                //ValidatingReaders Schema collection.
                oXmlValidatingReader.Schemas.Add (oXmlSchemaCollection);
                oXmlValidatingReader.ValidationType = ValidationType.Auto;

                //Attaching the event handler now in case of failures
                oXmlValidatingReader.ValidationEventHandler += new ValidationEventHandler
                (ValidationFailed);
            }
        }
    }
}

```

```
        //Actually validating the data in the XML file with a empty while.
        //which would fire the event ValidationEventHandler and invoke
        //our ValidationFailed function
        while (oXmlValidatingReader.Read())
        {
            //Note:- If any issue is faced in the above while it will invoke
ValidationFailed
            //which will in turn set the module level IsValid boolean variable to false
            //thus returning false as a signal to the calling function that the
ValidateXMLFile
            //function(this function) has encountered failure
            return IsValid;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Problem with clsXMLValidator " +
                "\r\n\r\n\r\n" + ex.Message,
                "XML Validation ERROR", MessageBoxButtons.OK,
                MessageBoxIcon.Error);
            return false;
        }
        finally
        {
            // close the readers, no matter what.
            oXmlValidatingReader.Close();
            oXmlTextReader.Close();
        }
    }
#endregion
#region Private Methods
/// <summary>
/// Event handler that occurs when there is a problem validating the
/// XML file against the XmlValidatingReader
/// </summary>
/// <param name="sender">The source object that raised the event</param>
/// <param name="e">The event arguments</param>
private void ValidationFailed (object sender, ValidationEventArgs args)
{
    IsValid = false;
    MessageBox.Show("Problem with clsXMLValidator " +
        "\r\n\r\n\r\n" + "Invalid XML File: " + args.Message,
        "XML Validation ERROR", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}
#endregion
}
#endregion
}
```