

```

using System;
using System.Collections;
using System.Windows.Forms;
using System.Xml;
using System.IO;
using System.Reflection;

namespace SB54_CSAI.TrackObjects
{
    #region XMLValidator CLASS
    /// <summary>
    /// Provides mechanisms for reading and writing XML files in association with the
    /// <see cref="SB54_CSAI.MediaPlayerControl.uctMediaPlayer">Media Player tracklist </
    see>
    /// and the <see cref="SB54_CSAI.ClientTrackList.uctClientTrackList">Client track lists
    </see>
    /// Save / Load command
    /// </summary>
    public class XMLReadWrite
    {
        #region Instance Fields
        //Instance Fields
        private ArrayList trackReadIn=new ArrayList();
        #endregion
        #region Public Constructor
        /// <summary>
        /// Simply constructs a new XMLReadWrite object
        /// provided
        /// </summary>
        public XMLReadWrite()
        {
        }
        #endregion
        #region Public Methods
        /// <summary>
        /// Uses the <see cref="SB54_CSAI.MediaPlayerControl.XMLValidator">XMLValidator </
    see>
        /// to validate the input filename parameter. If the file is not valid no further
    parsing
        /// is conducted, however if the file is valid, all Track XML elements will be used
    to
        /// try and create new track list items out of. The data contained in the Track XML
    elements
        /// must also be valid, the addReadXMLFile(string filename) method assists here.
        /// </summary>
        /// <param name="filename">The input XML file URL</param>
        public void ReadXMLTrackList(string filename)
        {
            XmlTextReader oXmlTextReader=null;
            try
            {
                //validate the Input file against the XML Schema
                trackReadIn.Clear();
                string sStartupPath = Application.StartupPath;
                XMLValidator oXMLValidator = new XMLValidator(@filename,
                    sStartupPath + @"\ReMP3TrackListSchema.xsd");
                //if not valid, stop processing
                if ( ! oXMLValidator.ValidateXMLFile() )
                {
                    MessageBox.Show("Problem with ReadXMLTrackList " +
                        "\r\n\r\n\r\n" + "The supplied file " + filename + " does not match
    the Schema",
                        "XML Tracklist Reading ERROR", MessageBoxButtons.OK,
                        MessageBoxIcon.Error);
                    return;
                }
                //is valid, so start the reading of the file
                oXmlTextReader = new XmlTextReader(@filename);
                //parse each file
                string eName="";
                while ( oXmlTextReader.Read() )

```

```

    {
        //parse XML nodes
        switch (oXmlTextReader.NodeType)
        {
            //parse XML elements
            case XmlNodeType.Element:
                eName=oXmlTextReader.Name;
                break;
            case XmlNodeType.Text:
                switch(eName)
                {
                    //only interesting in <Track> </Track> elements
                    case "Track":
                        addReadXMLFile(oXmlTextReader.Value);
                        break;
                }
                break;
        }
    }
}
catch(Exception ex)
{
    MessageBox.Show("Problem with ReadXMLTrackList " +
        "\r\n\r\n\r\n" + ex.Message,
        "XML Tracklist Reading ERROR", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}
finally
{
    if (oXmlTextReader != null)
    {
        //tidy up resources
        oXmlTextReader.Close();
    }
}
}

/// <summary>
/// Returns an array of string of all valid file URL's read from the input XML
/// file
/// </summary>
/// <returns>Returns an array of string of all valid file URL's read from the input
XML
/// file</returns>
public string[] getXMLReadFiles
{
    get
    {
        return trackReadIn.ToArray(Type.GetType("System.String")) as string[];
    }
}

/// <summary>
/// Saves all the current Media Player tracklist tracks to an XML file at
/// the URL specified by the filename parameter
/// </summary>
/// <param name="filename">The output XML file URL</param>
/// <param name="enTracks">An IEnumerable which holds all the track URLs for the
/// current tracklist in the <see cref="SB54_CSAI.MediaPlayerControl.uctMediaPlayer"
>
/// Media Player</see> and <see cref="SB54_CSAI.ClientTrackList.uctClientTrackList">
/// Client track lists </see></param>
public void WriteXMLTrackList(string filename, IEnumerable enTracks)
{
    XmlTextWriter oXmlTextWriter =null;
    try
    {
        //XmlTextWriter is used below. This class helps us to write xml on
        //many places like stream, console,file etc we are pushing the xml
        //on a file. Simply nest the Start & End method and its done
        oXmlTextWriter = new XmlTextWriter(@filename,null);
        oXmlTextWriter.Formatting = Formatting.Indented;
        oXmlTextWriter.WriteStartDocument();
    }
}

```

```
oXmlTextWriter.WriteStartElement("Tracklist");
//declare trackItem, will be filled by IEnumerator objects
MP3ListItem trackItem;
//loop through the IEnumerator of tracks, and write each to the output XML
file
enTracks.Reset();
while ( enTracks.MoveNext())
{
    trackItem = (MP3ListItem)enTracks.Current;
    oXmlTextWriter.WriteStartElement("Track");
    oXmlTextWriter.WriteString(trackItem.URL.ToString());
    oXmlTextWriter.WriteEndElement();
}
oXmlTextWriter.WriteEndElement();
oXmlTextWriter.WriteEndDocument();
oXmlTextWriter.Flush();
oXmlTextWriter.Close();
//done, so tell user
MessageBox.Show("The following file has been successfully created " +
    "\r\n\r\n\r\n" + @filename,
    "XML Tracklist Writing", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
}
catch(Exception ex)
{
    MessageBox.Show("Problem with WriteXMLTrackList " +
        "\r\n\r\n\r\n" + ex.Message,
        "XML Tracklist Writing ERROR", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}
finally
{
    if (oXmlTextWriter != null)
    {
        //tidy up resources
        oXmlTextWriter.Close();
    }
}
}
#endregion
#region Private Methods
/// <summary>
/// Takes a file URL, and will check that the file exists. If it does exist, the
/// file URL will be added to the internal collection. This method is to protect
/// the <see cref="SB54_CSAI.MediaPlayerControl.uctMediaPlayer">Media Player
tracklists </see>
/// and <see cref="SB54_CSAI.ClientTrackList.uctClientTrackList">Client track lists
</see>
/// against bogus information. Only valid files will make there way back to the
track list.
/// The Media Player itself will filter out any non audio tracks.
/// </summary>
/// <param name="filename">The name of the file to check</param>
private void addReadXMLFile(string filename)
{
    try
    {
        //If the file exists, add it to the internal collection
        FileInfo fi = new FileInfo(filename);
        if (fi.Exists)
        {
            trackReadIn.Add(filename);
        }
    }
    catch(Exception ex)
    {
        MessageBox.Show("Problem with addReadXMLFile " +
            "\r\n\r\n\r\n" + ex.Message,
            "XML Tracklist Reading ERROR", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
    }
}
}
#endregion
```

```
    }  
    #endregion  
}
```