Opening all files when the QASolution project loads

private void QADashboard\_Load(object sender, EventArgs e)

{

this.ControlBox = false;

if (!filesLoaded)

{

QADataModelLib.AccessData.[openAllFiles](#openAllFilesMethod)();

QAFileNameScoresModel.[loadQANameScoreDictionary()](#loadQANameScoreDictionaryMethod);

//SubjectNodesListModel.loadSubjectNodesList();

QADataModelLib.NodeChildrenDictionaryModel.[loadNodeChildrenDictionary()](#loadNodeChildrenDictionaryMethod);

QADataModelLib.TreeViewDictionaryModel.[loadTreeViewDictionary();](#loadTreeViewDictionaryMethod)

filesLoaded = true;

QACumulativeResultsModel.importQACumulativeResultsFile();

}

}

- - - - - -

QADataModelLib.AccessData.QADataModelLib.AccessData.openAllFiles();

[loadQACumulativeResultFile](#loadQACumulativeResultFileMethod)();

- - - - - -

QADataModelLib.AccessData.QADataModelLib.AccessData.private static void loadQACumulativeResultFile()

// Loads all the cumulative results data into qaCumulativeResultsTabl

- - - - - -

QAFileNameScoresModel.loadQANameScoreDictionary();

This method is called by the QADashboard\_Load Method, Its Purpose is to load the QAFileNameScores.txt file into the QANameScoreDictionary. It also sets currentMaxQAFileID,

public static void loadQANameScoreDictionary()

{

// Create a List of string to hold the lines in the input file

List<string> inputList = new List<string>();

//

if (File.Exists(qaNameScoreFilePath))

/\* C:\\Users\\Owner\\OneDrive\\Documents\\Learning\\\_CSharpQAFiles\\AccessoryFiles\\QAFileNameScores.txt \*/

{

//determine if there are data in the file and if so read it into the dictionary

var fil = new FileInfo(qaNameScoreFilePath);

long length = fil.Length;

if (length != 0)

{

string line;

int counter = 0;

// Read the file and enter it line by line into inputList

System.IO.StreamReader file =

new System.IO.StreamReader(qaNameScoreFilePath);

while ((line = file.ReadLine()) != null)

{

inputList.Add(line);

counter++;

}

file.Close();

// For each line in inputList parse it into the dictionary

var qaNameScoreDictionaryList = new List<string>(inputList);

//string line = "";

for (int i = 0; i < counter; i++)

{

line = inputList[i];

string[] keyAndValue = line.Split('~');

string keyString = keyAndValue[0];

int key = Int32.Parse(keyString);

if (key > currentMaxQAFileID)

{

currentMaxQAFileID = key;

}

string value = keyAndValue[1];

QANameScoreDictionary.Add(key, value);

}// End for loop parsing lines in inputList into QANameScoreDictionar

}// End if there are data in the file and if so read it into the dictionary

}// End if qaNameScoreFilePath FileExixts

}// End setQANameScoreDictionary

- - - - - -

- - - - - -

QADataModelLib.NodeChildrenDictionaryModel.loadNodeChildrenDictionary();

/\* This method determines if there data in the nodeChildDirectory and if

there are it splits each line in the file (on’^’) into a key and a

value which it then loads into the nodeChildrenDictionary \*/

{ // Dictionary<string, int> nodeChildrenDictionary = new Dictionary<string, int>();

if (File.Exists(nodeChildDictionaryPath))

**// nodeChildDictionaryPath = C:\\Users\\Owner\\OneDrive\\Documents\\Learning\\\_CSharpQAFiles\\AccessoryFiles\\NodeChildDictionary.txt**

{

//If length of the file is not 0 then there are data in the file and if so read it into the dictionary

var fil = new FileInfo(nodeChildDictionaryPath);

long length = fil.Length;

if (length != 0)

{

// Read in the file and parse it into the dictionary

var logFile = File.ReadAllLines(nodeChildDictionaryPath);

var nodeChildrenDictionaryList = new List<string>(logFile);

foreach (string line in nodeChildrenDictionaryList)

{

string[] keyAndValue = line.Split('^');

int result = Int32.Parse(keyAndValue[1]);

nodeChildrenDictionary.Add(keyAndValue[0], result);

}

}

}

}// End loadNodeChildrenDictionary

- - - - - -

/// <summary>

/// This method is called getTreeViewDictionary

/// It opens reads the lines in the TreeViewDictionary.txt into a ArrayList

/// and extracts the '^' keys and values

/// and creates the TreeViewDictionary<string,string> from them

/// </summary>

public static void loadTreeViewDictionary()

{

var treeViewdictionaryList = new List<string>();

if (File.Exists(treeViewDictionaryPath))

**/\* treeViewDictionaryPath = C:\\Users\\Owner\\OneDrive\\Documents\\Learning\\\_CSharpQAFiles\\AccessoryFiles\\TreeViewDictionary.txt" \*/**

{

//determine if there are data in the file and if so read it into the dictionary

var fil = new FileInfo(treeViewDictionaryPath);

long length = fil.Length;

if (length != 0)

{

// Read in the file and parse it into the dictionary

var logFile = File.ReadAllLines(treeViewDictionaryPath);

treeViewdictionaryList = new List<string>(logFile);

//logFile.toList<string>;

foreach (string line in treeViewdictionaryList)

{

string[] keyAndValue = line.Split('^');

TreeViewDictionary.Add(keyAndValue[0], keyAndValue[1]);

}

}

}// End if(File.Exists

}// loadTreeViewDictionary