Unit – тесты

Тест №1

[TestClass]

public class UnitTest1

{

String currentGroup = "TestGroup";

bool isGroupsMode = false;

String fileName = "Names.txt";

int index = 0;

int amount = 3;

List<String> testFileData = new List<String>

{

"Данил",

"Саша",

"Таня"

};

String testName = "TestGroup";

int testMinAge = 3;

int testMaxAge = -1;

int testMaxAmount = -1;

int testAge = -1;

[TestMethod]

public void Test\_OpenClouseButton\_Click()

{

StreamReader streamReader = new StreamReader("Data\\" + currentGroup + "\\Names.txt", true);

StreamReader streamReader1 = new StreamReader("Data\\" + currentGroup + "\\Age.txt", true);

StreamReader streamReader2 = new StreamReader("Data\\" + currentGroup + "\\BDD.txt", true);

String name;

String age;

String bdd;

while (true)

{

name = streamReader.ReadLine();

if (name == null)

{

break;

}

age = streamReader1.ReadLine();

bdd = streamReader2.ReadLine();

Assert.IsNotNull(name);

Assert.IsNotNull(age);

Assert.IsNotNull(bdd);

}

streamReader.Close();

streamReader1.Close();

streamReader2.Close();

}

[TestMethod]

public void Test\_Init()

{

bool successful = true;

try

{

Directory.CreateDirectory("Data\\");

FileStream fileStreamOfNames = null;

FileStream fileStreamOfCourses = null;

FileStream fileStreamOfMin = null;

FileStream fileStreamOfMax = null;

FileStream fileStreamOfMaxA = null;

FileInfo fileInfo = new FileInfo("Data\\Names.txt");

if (!fileInfo.Exists)

{

fileStreamOfNames = fileInfo.Create();

fileStreamOfNames.Close();

}

FileInfo fileInfo1 = new FileInfo("Data\\Course.txt");

if (!fileInfo1.Exists)

{

fileStreamOfCourses = fileInfo1.Create();

fileStreamOfCourses.Close();

}

FileInfo fileInfo2 = new FileInfo("Data\\Min.txt");

if (!fileInfo2.Exists)

{

fileStreamOfMin = fileInfo2.Create();

fileStreamOfMin.Close();

}

FileInfo fileInfo3 = new FileInfo("Data\\Max.txt");

if (!fileInfo3.Exists)

{

fileStreamOfMax = fileInfo3.Create();

fileStreamOfMax.Close();

}

FileInfo fileInfo4 = new FileInfo("Data\\MaxA.txt");

if (!fileInfo4.Exists)

{

fileStreamOfMaxA = fileInfo4.Create();

fileStreamOfMaxA.Close();

}

}

catch

{

successful = false;

}

Assert.IsTrue(successful);

StreamReader streamWriterOfNames = new StreamReader("Data\\Names.txt", true);

StreamReader streamWriterOfCourses = new StreamReader("Data\\Course.txt", true);

StreamReader streamWriterOfMin = new StreamReader("Data\\Min.txt", true);

StreamReader streamWriterOfMax = new StreamReader("Data\\Max.txt", true);

StreamReader streamWriterOfMaxA = new StreamReader("Data\\MaxA.txt", true);

while (true)

{

string groupName = streamWriterOfNames.ReadLine();

if (groupName == null)

{

break;

}

string curse = streamWriterOfCourses.ReadLine();

string min = streamWriterOfMin.ReadLine();

string max = streamWriterOfMax.ReadLine();

int counter = 0;

string maxA = streamWriterOfMaxA.ReadLine();

Assert.IsNotNull(groupName);

Assert.IsNotNull(curse);

Assert.IsNotNull(min);

Assert.IsNotNull(max);

Assert.IsNotNull(maxA);

using (StreamReader streamReader = new StreamReader("Data\\" + groupName + "\\Names.txt"))

{

while (streamReader.ReadLine() != null)

{

counter++;

}

}

Assert.AreEqual(counter, amount);

}

streamWriterOfNames.Close();

streamWriterOfCourses.Close();

streamWriterOfMin.Close();

streamWriterOfMax.Close();

streamWriterOfMaxA.Close();

}

[TestMethod]

public void Test\_LimitCheck()

{

bool enabled;

StreamReader streamReader = new StreamReader("Data\\" + currentGroup + "\\Names.txt", true);

StreamReader streamReaderForNames = new StreamReader("Data\\Names.txt", true);

StreamReader streamReaderForMaxAmount = new StreamReader("Data\\MaxA.txt", true);

int childCounter = 0;

int indexCounter = 0;

int localCounter = 0;

String line;

String name;

int maxAmountOfChildren;

while (true)

{

name = streamReader.ReadLine();

if (name == null)

{

break;

}

childCounter++;

}

while (true)

{

line = streamReaderForNames.ReadLine();

if (line == currentGroup)

{

break;

}

indexCounter++;

}

while (true)

{

maxAmountOfChildren = Convert.ToInt32(streamReaderForMaxAmount.ReadLine());

if (localCounter == indexCounter)

{

break;

}

localCounter++;

}

if (maxAmountOfChildren == childCounter)

{

enabled = false;

}

else

{

enabled = true;

}

Assert.IsTrue(!enabled);

streamReader.Close();

streamReaderForNames.Close();

streamReaderForMaxAmount.Close();

}

[TestMethod]

public void Test\_ReadAllData()

{

List<String> fileData = new List<String>();

String path;

int counter = 0;

if (isGroupsMode)

{

path = "Data\\" + fileName;

}

else

{

path = "Data\\" + currentGroup + "\\" + fileName;

}

using (StreamReader streamReader = new StreamReader(path))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (!(counter == index))

{

fileData.Add(line);

}

line = streamReader.ReadLine();

counter++;

}

}

Assert.IsNotNull(fileData);

}

[TestMethod]

public void Test\_WriteAllData()

{

List<String> fileData = new List<String>();

String path;

int counter = 0;

if (this.isGroupsMode)

{

path = "Data\\" + fileName;

}

else

{

path = "Data\\" + currentGroup + "\\" + fileName;

}

using (StreamWriter streamWriter = new StreamWriter(path))

{

foreach (string item in testFileData)

{

streamWriter.WriteLine(item);

}

}

using (StreamReader streamReader = new StreamReader(path))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (!(counter == index))

{

fileData.Add(line);

}

line = streamReader.ReadLine();

counter++;

}

}

Assert.IsNotNull(fileData);

}

[TestMethod]

public void Test\_GroupDataCheck()

{

bool nameIsCorrect = true;

bool minOrMaxNotZero = true;

bool maxMoreThenMin = true;

bool maxANotZero = true;

using (StreamReader streamReader = new StreamReader("Data\\Names.txt"))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (line == testName)

{

nameIsCorrect = false;

}

line = streamReader.ReadLine();

}

}

if (testMaxAge < 0 || testMinAge < 0)

{

minOrMaxNotZero = false;

}

if (testMinAge > testMaxAge)

{

maxMoreThenMin = false;

}

if (testMaxAmount < 1)

{

maxANotZero = false;

}

Assert.AreEqual(nameIsCorrect, false);

Assert.AreEqual(minOrMaxNotZero, false);

Assert.AreEqual(maxMoreThenMin, false);

Assert.AreEqual(maxANotZero, false);

}

[TestMethod]

public void Test\_ChildDataCheck()

{

int minAge;

int maxAge;

int counter = 0;

bool ageMoreThemZero = true;

bool ageInRange = true;

if (testAge < 0)

{

ageMoreThemZero = false;

}

using (StreamReader sr = new StreamReader("Data\\Names.txt"))

{

string line = sr.ReadLine();

while (line != currentGroup)

{

line = sr.ReadLine();

counter++;

}

}

using (StreamReader sr = new StreamReader("Data\\Min.txt"))

{

int localCounter = 0;

string line = sr.ReadLine();

while (localCounter != counter)

{

line = sr.ReadLine();

localCounter++;

}

minAge = Convert.ToInt32(line);

}

using (StreamReader sr = new StreamReader("Data\\Max.txt"))

{

int localCounter = 0;

string line = sr.ReadLine();

while (localCounter != counter)

{

line = sr.ReadLine();

localCounter++;

}

maxAge = Convert.ToInt32(line);

}

if (testAge < minAge || testAge > maxAge)

{

ageInRange = false;

}

Assert.AreEqual(false, ageMoreThemZero);

Assert.AreEqual(false, ageInRange);

}

}

Тест №2

[TestClass]

public class UnitTest2

{

String currentGroup = "TestGroup1";

bool isGroupsMode = true;

String fileName = "Names.txt";

int index = 0;

int amount = 3;

List<String> testFileData = new List<String>

{

"TestGroup",

"TestGroup1",

};

String testName = "TestGroup2";

int testMinAge = 2;

int testMaxAge = 3;

int testMaxAmount = 5;

int testAge = 3;

[TestMethod]

public void Test\_OpenClouseButton\_Click()

{

StreamReader streamReader = new StreamReader("Data\\" + currentGroup + "\\Names.txt", true);

StreamReader streamReader1 = new StreamReader("Data\\" + currentGroup + "\\Age.txt", true);

StreamReader streamReader2 = new StreamReader("Data\\" + currentGroup + "\\BDD.txt", true);

String name;

String age;

String bdd;

while (true)

{

name = streamReader.ReadLine();

if (name == null)

{

break;

}

age = streamReader1.ReadLine();

bdd = streamReader2.ReadLine();

Assert.IsNotNull(name);

Assert.IsNotNull(age);

Assert.IsNotNull(bdd);

}

streamReader.Close();

streamReader1.Close();

streamReader2.Close();

}

[TestMethod]

public void Test\_Init()

{

bool successful = true;

try

{

Directory.CreateDirectory("Data\\");

FileStream fileStreamOfNames = null;

FileStream fileStreamOfCourses = null;

FileStream fileStreamOfMin = null;

FileStream fileStreamOfMax = null;

FileStream fileStreamOfMaxA = null;

FileInfo fileInfo = new FileInfo("Data\\Names.txt");

if (!fileInfo.Exists)

{

fileStreamOfNames = fileInfo.Create();

fileStreamOfNames.Close();

}

FileInfo fileInfo1 = new FileInfo("Data\\Course.txt");

if (!fileInfo1.Exists)

{

fileStreamOfCourses = fileInfo1.Create();

fileStreamOfCourses.Close();

}

FileInfo fileInfo2 = new FileInfo("Data\\Min.txt");

if (!fileInfo2.Exists)

{

fileStreamOfMin = fileInfo2.Create();

fileStreamOfMin.Close();

}

FileInfo fileInfo3 = new FileInfo("Data\\Max.txt");

if (!fileInfo3.Exists)

{

fileStreamOfMax = fileInfo3.Create();

fileStreamOfMax.Close();

}

FileInfo fileInfo4 = new FileInfo("Data\\MaxA.txt");

if (!fileInfo4.Exists)

{

fileStreamOfMaxA = fileInfo4.Create();

fileStreamOfMaxA.Close();

}

}

catch

{

successful = false;

}

Assert.IsTrue(successful);

StreamReader streamWriterOfNames = new StreamReader("Data\\Names.txt", true);

StreamReader streamWriterOfCourses = new StreamReader("Data\\Course.txt", true);

StreamReader streamWriterOfMin = new StreamReader("Data\\Min.txt", true);

StreamReader streamWriterOfMax = new StreamReader("Data\\Max.txt", true);

StreamReader streamWriterOfMaxA = new StreamReader("Data\\MaxA.txt", true);

while (true)

{

string groupName = streamWriterOfNames.ReadLine();

if (groupName == null)

{

break;

}

string curse = streamWriterOfCourses.ReadLine();

string min = streamWriterOfMin.ReadLine();

string max = streamWriterOfMax.ReadLine();

int counter = 0;

string maxA = streamWriterOfMaxA.ReadLine();

Assert.IsNotNull(groupName);

Assert.IsNotNull(curse);

Assert.IsNotNull(min);

Assert.IsNotNull(max);

Assert.IsNotNull(maxA);

using (StreamReader streamReader = new StreamReader("Data\\" + groupName + "\\Names.txt"))

{

while (streamReader.ReadLine() != null)

{

counter++;

}

}

Assert.AreEqual(counter, amount);

}

streamWriterOfNames.Close();

streamWriterOfCourses.Close();

streamWriterOfMin.Close();

streamWriterOfMax.Close();

streamWriterOfMaxA.Close();

}

[TestMethod]

public void Test\_LimitCheck()

{

bool enabled;

StreamReader streamReader = new StreamReader("Data\\" + currentGroup + "\\Names.txt", true);

StreamReader streamReaderForNames = new StreamReader("Data\\Names.txt", true);

StreamReader streamReaderForMaxAmount = new StreamReader("Data\\MaxA.txt", true);

int childCounter = 0;

int indexCounter = 0;

int localCounter = 0;

String line;

String name;

int maxAmountOfChildren;

while (true)

{

name = streamReader.ReadLine();

if (name == null)

{

break;

}

childCounter++;

}

while (true)

{

line = streamReaderForNames.ReadLine();

if (line == currentGroup)

{

break;

}

indexCounter++;

}

while (true)

{

maxAmountOfChildren = Convert.ToInt32(streamReaderForMaxAmount.ReadLine());

if (localCounter == indexCounter)

{

break;

}

localCounter++;

}

if (maxAmountOfChildren == childCounter)

{

enabled = false;

}

else

{

enabled = true;

}

Assert.IsTrue(enabled);

streamReader.Close();

streamReaderForNames.Close();

streamReaderForMaxAmount.Close();

}

[TestMethod]

public void Test\_ReadAllData()

{

List<String> fileData = new List<String>();

String path;

int counter = 0;

if (isGroupsMode)

{

path = "Data\\" + fileName;

}

else

{

path = "Data\\" + currentGroup + "\\" + fileName;

}

using (StreamReader streamReader = new StreamReader(path))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (!(counter == index))

{

fileData.Add(line);

}

line = streamReader.ReadLine();

counter++;

}

}

Assert.IsNotNull(fileData);

}

[TestMethod]

public void Test\_WriteAllData()

{

List<String> fileData = new List<String>();

String path;

int counter = 0;

if (this.isGroupsMode)

{

path = "Data\\" + fileName;

}

else

{

path = "Data\\" + currentGroup + "\\" + fileName;

}

using (StreamWriter streamWriter = new StreamWriter(path))

{

foreach (string item in testFileData)

{

streamWriter.WriteLine(item);

}

}

using (StreamReader streamReader = new StreamReader(path))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (!(counter == index))

{

fileData.Add(line);

}

line = streamReader.ReadLine();

counter++;

}

}

Assert.IsNotNull(fileData);

}

[TestMethod]

public void Test\_GroupDataCheck()

{

bool nameIsCorrect = true;

bool minOrMaxNotZero = true;

bool maxMoreThenMin = true;

bool maxANotZero = true;

using (StreamReader streamReader = new StreamReader("Data\\Names.txt"))

{

string line = streamReader.ReadLine();

while (line != null)

{

if (line == testName)

{

nameIsCorrect = false;

}

line = streamReader.ReadLine();

}

}

if (testMaxAge < 0 || testMinAge < 0)

{

minOrMaxNotZero = false;

}

if (testMinAge > testMaxAge)

{

maxMoreThenMin = false;

}

if (testMaxAmount < 1)

{

maxANotZero = false;

}

Assert.AreEqual(nameIsCorrect, true);

Assert.AreEqual(minOrMaxNotZero, true);

Assert.AreEqual(maxMoreThenMin, true);

Assert.AreEqual(maxANotZero, true);

}

[TestMethod]

public void Test\_ChildDataCheck()

{

int minAge;

int maxAge;

int counter = 0;

bool ageMoreThemZero = true;

bool ageInRange = true;

if (testAge < 0)

{

ageMoreThemZero = false;

}

using (StreamReader sr = new StreamReader("Data\\Names.txt"))

{

string line = sr.ReadLine();

while (line != currentGroup)

{

line = sr.ReadLine();

counter++;

}

}

using (StreamReader sr = new StreamReader("Data\\Min.txt"))

{

int localCounter = 0;

string line = sr.ReadLine();

while (localCounter != counter)

{

line = sr.ReadLine();

localCounter++;

}

minAge = Convert.ToInt32(line);

}

using (StreamReader sr = new StreamReader("Data\\Max.txt"))

{

int localCounter = 0;

string line = sr.ReadLine();

while (localCounter != counter)

{

line = sr.ReadLine();

localCounter++;

}

maxAge = Convert.ToInt32(line);

}

if (testAge < minAge || testAge > maxAge)

{

ageInRange = false;

}

Assert.AreEqual(true, ageMoreThemZero);

Assert.AreEqual(true, ageInRange);

}

}

Результаты тестирования

