Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Кубанский Государственный Технологический Университет»

Кафедра информационных систем и программирования

Отчет по лабораторной работе №7

По дисциплине «Рефакторинг и работа с унаследованным кодом»

«Рефакторинг перед добавлением функциональности, схожей с уже существующей»

Выполнил студент

Группы 21-КБ-ПР2

Жулькин Д.П.

Преподаватель:

Степанова Елизавета Владимировна

Краснодар

2024

Цель работы: изучить процесс рефакторинга перед добавлением новой функциональности, схожей с уже существующей.

Задание:

1) Преобразовать процесс парсинга файлов в соответствии с предложенными в пункте 2.1 изменениями, применив описанные в теории методы рефакторинга.

2) Добавить поддержку нового формата входных файлов.

3) Проверить согласованность результатов и корректность работы нового варианта программы с результатами старой версии.

4) Оформить отчёт.

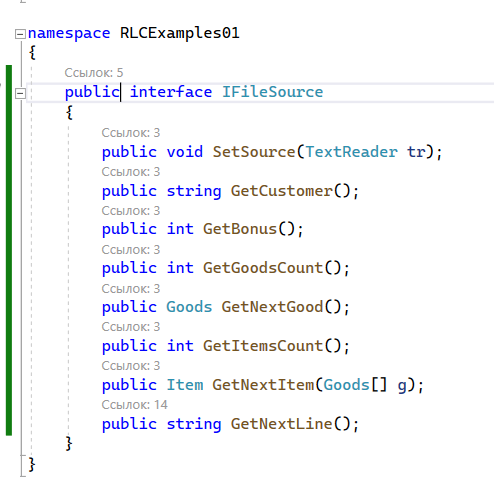


Рисунок 1 – Создание интерфейса IFileSource

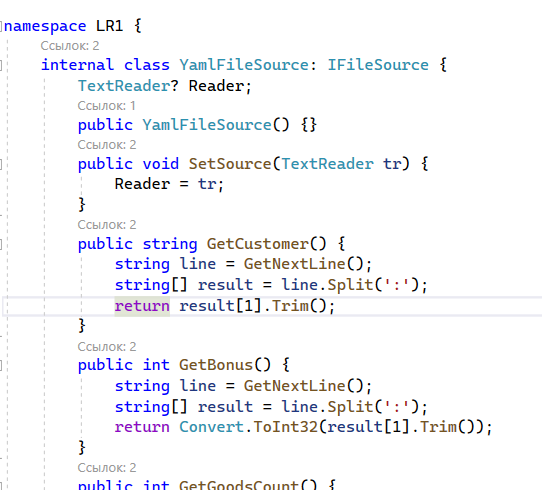


Рисунок 2 – Переименование класса и добавление интерфейса

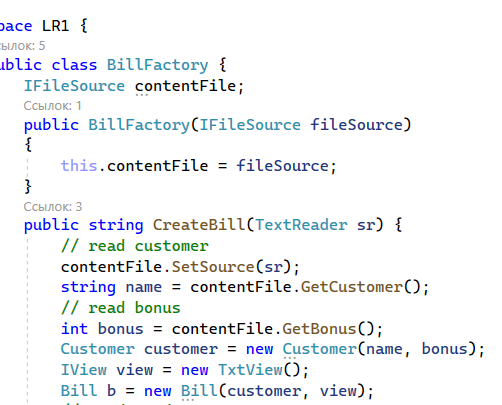


Рисунок 3 – Параметризация класс BillFactory

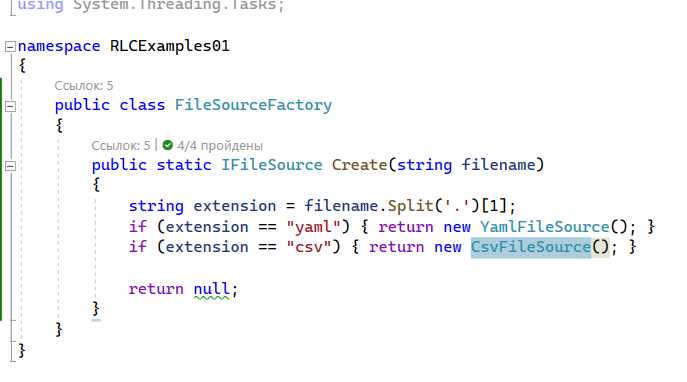


Рисунок 4 – Новый класс FileSourceFactory

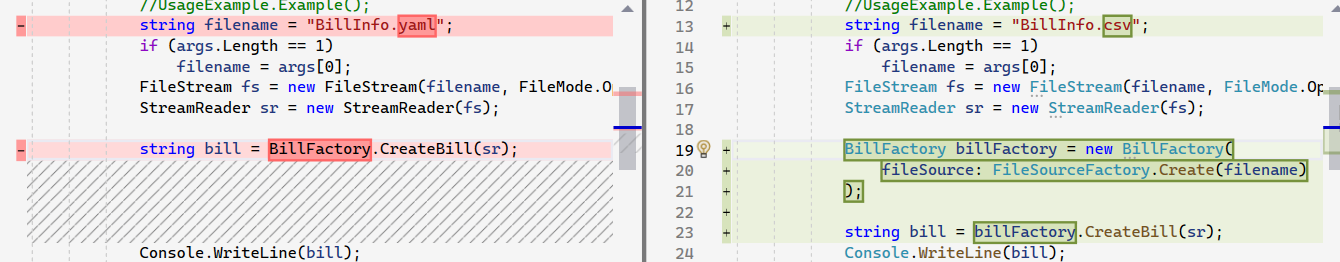


Рисунок 5 – Функция Main

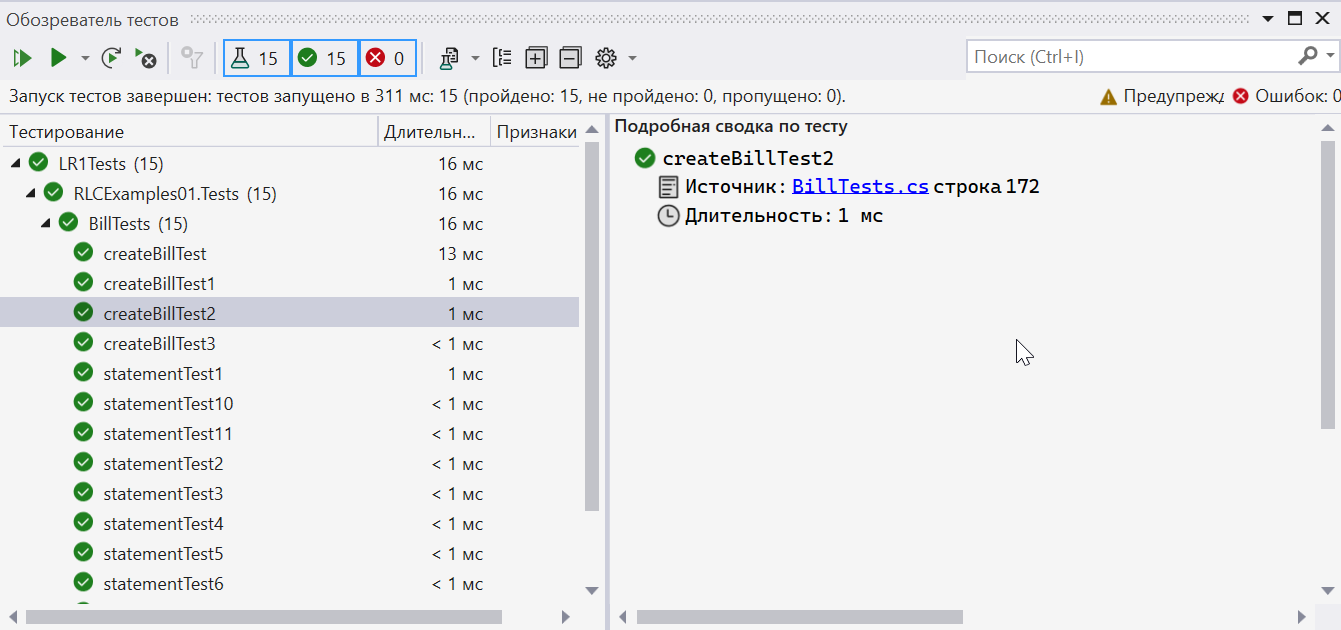


Рисунок 4 – Тесты выполняются успешно

Program.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.IO;

using RLCExamples01;

using LR1;

namespace RLCExamples01 {

public class Program {

static void Main(string[] args)

{

//UsageExample.Example();

string filename = "BillInfo.csv";

if (args.Length == 1)

filename = args[0];

FileStream fs = new FileStream(filename, FileMode.Open);

StreamReader sr = new StreamReader(fs);

BillFactory billFactory = new BillFactory(

fileSource: FileSourceFactory.Create(filename)

);

string bill = billFactory.CreateBill(sr);

Console.WriteLine(bill);

}

}

}

FileSourceFactory.cs

using LR1;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RLCExamples01

{

public class FileSourceFactory

{

public static IFileSource Create(string filename)

{

string extension = filename.Split('.')[1];

if (extension == "yaml") { return new YamlFileSource(); }

if (extension == "csv") { return new CsvFileSource(); }

return null;

}

}

}

IFileSource.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RLCExamples01

{

public interface IFileSource

{

public void SetSource(TextReader tr);

public string GetCustomer();

public int GetBonus();

public int GetGoodsCount();

public Goods GetNextGood();

public int GetItemsCount();

public Item GetNextItem(Goods[] g);

public string GetNextLine();

}

}

CsvFileSource.cs

using LR1;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RLCExamples01

{

internal class CsvFileSource: IFileSource

{

TextReader? Reader;

public CsvFileSource() { }

public void SetSource(TextReader tr) {

Reader = tr;

}

public string GetCustomer()

{

string line = GetNextLine();

string[] result = line.Split(',');

return result[1].Trim();

}

public int GetBonus()

{

string line = GetNextLine();

string[] result = line.Split(',');

return Convert.ToInt32(result[1].Trim());

}

public int GetGoodsCount()

{

string line = GetNextLine();

string[] result = line.Split(",");

return Convert.ToInt32(result[1].Trim());

}

public Goods GetNextGood()

{

string line = GetNextLine();

string[] result = line.Split(',');

string type = result[2].Trim();

return GoodsFactory.Create(type, result[1]);

}

public int GetItemsCount()

{

string[] result = GetNextLine().Split(',');

return Convert.ToInt32(result[1].Trim());

}

public Item GetNextItem(Goods[] g)

{

string[] result = GetNextLine().Split(',');

int gid = Convert.ToInt32(result[1].Trim());

double price = Convert.ToDouble(result[2].Trim());

int qty = Convert.ToInt32(result[3].Trim());

return new Item(g[gid - 1], qty, price);

}

public string GetNextLine()

{

string line = Reader.ReadLine();

while (line.StartsWith("#"))

{

line = Reader.ReadLine();

}

return line;

}

}

}