**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «ЯПВУ»

на тему: «Делегаты и события»

Выполнил: студент гр. ИП-22

Козловский В.В.

Принял: ассистент

Ракицкий А.А.

Гомель 2020

**Цель**: получить навыки использования делегатов и событий.

**Задание.**

**Вариант 10**

1. Подготовить текстовый файл, содержащий информацию о хранящемся на складе товаре: Название товара, поставщик, наценка в процентах, количество товара, поступившего на склад (разделителем в файле служит точка с запятой). Подготовить текстовый файл, содержащий информацию о товаре: название товара, цена.
2. Класс **Склад** должен содержать следующие элементы: поля для хранения названия товара, поставщика, отпускной цены (исходная цена + наценка), количества товара, свойство для определения стоимости товара, метод для вывода информации, методы для реализации шаблона «наблюдатель» (см. ниже).
3. Класс **Товар** должен содержать следующие элементы: поля для хранения названия товара, цены, методы для реализации шаблона «наблюдатель» (см. ниже).
4. Разработать программу, которая выполняет следующие действия:

* Считывает информацию из первого файла в массив объектов класса Склад.
* Считывает информацию из второго файла в массив объектов класса Товар.
* Выводит информацию о наличии на складе заданного товара в виде:

*ИП Петров В.В.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *№* | *Наименование товара* | *Цена* | *Наличие на складе* | *Стоимость* |
| *1* | *Стул* | *23000* | *200* | *4600000* |
| *2* | *Диван «Лагуна»* | *120000* | *15* | *35002000* |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *2* | *Шкаф* | *120000* | *25* | *43500* |
|  |  |  |  |  |
| *№* | *Наименование товара* | *Цена* | *Наличие на складе* | *Стоимость* |
| *1* | *Стул* | *25000* | *385* | *2800000* |

*ООО «Прогресс»*

1. Реализовать паттерн Наблюдатель
   1. Класс товар реализует объект наблюдателя
   2. Класс Склад реализует объект наблюдения

* На основе делегата - при изменении названия товара автоматически изменяется название товара на складе. Для этого каждый товар на складе должен быть «зарегистрирован» в соответствующем объекте класса **Товар**. Товар, для которого меняется название, выбирается пользователем.

На основе стандартной обработки событий при изменении цены на товар изменяется отпускная цена на складе для товаров, наценка на которые меньше 15%.

**Код программы:**

CollectionMapProducts.cs:

using CsvHelper.Configuration;

namespace Task1

{

public class CollectionMapProducts : ClassMap<Product>

{

public CollectionMapProducts()

{

Map(m => m.Name).Name(nameof(Product.Name));

Map(m => m.Price).Name(nameof(Product.Price));

}

}

}

CollectionMapStorage.cs:

using CsvHelper.Configuration;

namespace Task1

{

public class CollectionMapStorage : ClassMap<Storage>

{

public CollectionMapStorage()

{

Map(m => m.ProductName).Name(nameof(Storage.ProductName));

Map(m => m.Supplier).Name(nameof(Storage.Supplier));

Map(m => m.Markup).Name(nameof(Storage.Markup));

Map(m => m.SellingPrice).Name(nameof(Storage.SellingPrice));

Map(m => m.ProductsAmount).Name(nameof(Storage.ProductsAmount));

}

}

}

FileExtensions.cs:

|  |
| --- |
| using CsvHelper; |
|  | using System; |
|  | using System.IO; |
|  | using System.Linq; |
|  |  |
|  | namespace Task1 |
|  | { |
|  | public static class FileExtensions |
|  | { |
|  | public static Product[] GetProducts(string filePath) |
|  | { |
|  | try |
|  | { |
|  | using (var reader = new StreamReader(filePath)) |
|  | { |
|  | using (var csv = new CsvReader(reader, System.Globalization.CultureInfo.CurrentCulture)) |
|  | { |
|  | csv.Configuration.RegisterClassMap<CollectionMapProducts>(); |
|  | var records = csv.GetRecords<Product>(); |
|  | return records.ToArray(); |
|  | } |
|  | } |
|  | } |
|  | catch (Exception ex) |
|  | { |
|  | Console.WriteLine(ex.Message); |
|  | return null; |
|  | } |
|  | } |
|  |  |
|  | public static Storage[] GetStorage(string filePath) |
|  | { |
|  | try |
|  | { |
|  | using (var reader = new StreamReader(filePath)) |
|  | { |
|  | using (var csv = new CsvReader(reader, System.Globalization.CultureInfo.CurrentCulture)) |
|  | { |
|  | csv.Configuration.RegisterClassMap<CollectionMapStorage>(); |
|  | var records = csv.GetRecords<Storage>(); |
|  | return records.ToArray(); |
|  | } |
|  | } |
|  | } |
|  | catch (Exception ex) |
|  | { |
|  | Console.WriteLine(ex.Message); |
|  | return null; |
|  | } |
|  | } |
|  | } |
|  | } |

Product.cs:

using System;

namespace Task1

{

public class Product

{

private Action<string> NameHandler;

private event Action<decimal> PriceHandler;

private string \_name;

private decimal \_price;

public string Name

{

get => \_name;

set

{

\_name = value;

Notify(\_name);

}

}

public decimal Price

{

get => \_price;

set

{

\_price = value;

Notify(\_price);

}

}

public void AddObserver(Action<string> nameHandler)

{

NameHandler += nameHandler;

}

public void AddObserver(Action<decimal> priceHandler)

{

PriceHandler += priceHandler;

}

public void RemoveObserver(Action<string> nameHandler)

{

NameHandler -= nameHandler;

}

public void RemoveObserver(Action<decimal> priceHandler)

{

PriceHandler -= priceHandler;

}

public void Notify(string key)

{

NameHandler?.Invoke(key);

}

public void Notify(decimal obj)

{

PriceHandler?.Invoke(obj);

}

}

}

Program.cs:

using System;

namespace Task1

{

internal class Program

{

static void Main()

{

const string fileProduct = "Products.csv";

const string fileStorage = "Storage.csv";

Product[] products = FileExtensions.GetProducts(fileProduct);

Storage[] storages = FileExtensions.GetStorage(fileStorage);

products[0].AddObserver(storages[0].UpdateName);

products[0].AddObserver(storages[0].UpdateSellingPrice);

products[0].AddObserver(storages[3].UpdateName);

products[0].AddObserver(storages[3].UpdateSellingPrice);

products[1].AddObserver(storages[1].UpdateName);

products[1].AddObserver(storages[1].UpdateSellingPrice);

products[2].AddObserver(storages[2].UpdateName);

products[2].AddObserver(storages[2].UpdateSellingPrice);

foreach (var storage in storages)

{

Console.WriteLine(storage);

}

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

products[0].Name = "Table";

products[0].Price -= 1.55m;

foreach (var storage in storages)

{

Console.WriteLine(storage);

}

}

}

}

Storage.cs:

namespace Task1

{

public class Storage

{

public string ProductName { get; set; }

public string Supplier { get; set; }

public decimal Markup { get; set; }

public int ProductsAmount { get; set; }

public decimal SellingPrice { get; set; }

public decimal ProductsCost => SellingPrice \* ProductsAmount;

public void UpdateName(string name)

{

ProductName = name;

}

public void UpdateSellingPrice(decimal price)

{

if (Markup < 15m)

{

SellingPrice = price + price \* Markup / 100;

}

}

public override string ToString() => ($"{ProductName};{SellingPrice,6:f2};{ProductsAmount};{ProductsCost,6:f2}");

}

}

**Результат выполнения программы:**

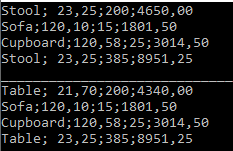


Рисунок 1 – Результат выполнения задания 1

**Вывод:** получил навыки использования делегатов и событий.