Практическая работа №5
Обработка и вызов событий

1Цель работы

1.1 Научиться создавать, вызывать и обрабатывать события на C#.

2 Литература

2.1 https://metanit.com/sharp/tutorial/3.14.php

2.2 https://docs.microsoft.com/ru-ru/dotnet/standard/events/how-to-raise-and-consume-events

3 Задание

3.1 Создание в классе события, срабатывающего при изменении значения свойства

3.1.1 Создать консольное приложение. Добавить в него класс Пользователь с закрытыми полями логин и пароль и открытыми свойствами на чтение и запись для доступа к этим полям.

3.1.2 Добавить в класс Пользователь:

- делегат типа void, принимающий строковое значение,
- событие типа созданного делегата.

3.1.3 Реализовать вызов события при изменении логина и пароля. В параметрах метода события должен передаваться текст «Логин изменен на *значение*» и «Пароль изменен на *значение*» в зависимости от того, в каком свойстве вызвано событие. Вместо слова *значение* должно отображаться присваиваемое значение.

3.1.4 Внести изменения в свойства класса:

- при попытке изменить логин на пустую строку или строку из пробельных символов в параметрах метода события должен передаваться текст «Невозможно изменить логин на пустую строку»,
- при указании пароля длиной меньше 6 и больше 20 символов в параметрах метода события должен передаваться текст «Длина пароля должна быть от 6 до 20 символов».

3.1.5 В консольном приложении создать объект класса пользователя и реализовать вывод на экран текста о изменении логина и пароля, используя подписку на событие.

Протестировать в консольном приложении работу с корректными и некорректными значениями для свойств.

3.2 Получение данных об отправителе события

3.2.1 Создать консольное приложение. Добавить в него класс Пользователь с закрытыми полями логин и пароль и открытыми свойствами на чтение и запись для доступа к этим полям.

3.2.2 Добавить в класс Пользователь:

- событие типа EventHandler.

3.2.3 Реализовать вызов события при изменении логина и пароля. В параметрах метода события должен передаваться текущий объект и EventArgs.Empty.

3.2.4 В консольном приложении создать объект класса пользователя и, используя подписку на событие, реализовать вывод на экран текста об изменении свойства класса в следующем формате:

Изменены данные пользователя со следующим логином: *логин*.

Логин должен браться из параметров объекта-отправителя. Для получения данных о пользователе sender нужно привести к типу Пользователь.

3.3 Создание класса данных события

3.3.1 Создать консольное приложение. Добавить в него класс Пользователь с закрытыми полями логин и пароль и открытыми свойствами на чтение и запись для доступа к этим полям.

3.3.2 Создать класс данных события DataEventArgs (наследник класса EventArgs) со следующими свойствами: параметр (строка) и дата (дата и время).

3.3.3 Добавить в класс Пользователь:

- событие типа EventHandler<DataEventArgs>.

3.3.4 Реализовать вызов события при изменении логина и пароля. В параметрах метода события должен передаваться текущий объект и объект типа DataEventArgs, у которого настроены свойства:

- параметр: «логин» или «пароль» (в зависимости от того, в каком свойстве вызывается событие)

- дата: текущая дата и время.

3.3.5 В консольном приложении создать объект класса пользователя и, используя подписку на событие, реализовать вывод на экран текста о изменении свойства класса в следующем формате:

дата и время: у пользователя *логин* изменено *свойство*.

Дата и время должны браться из параметров объекта класса данных события.

Свойство должно браться из параметров объекта класса данных события.

Логин должен браться из параметров объекта-отправителя.

3.4 Создание в форме события, срабатывающего при изменении текста в поле ввода другой формы

3.4.1 Создать оконное приложение из двух форм:

- первая отображает данные в табличном виде (источник данных: файлы из указанной пользователем папки, для указания источника данных настроить свойство DataSource у DataGridView),

- вторая используется для указания в поле ввода фильтра (части названия файла).

3.4.2 Реализовать открытие второй формы при нажатии на кнопку «Фильтр» на первой форме.

3.4.3 Реализовать фильтрацию данных на первой форме, используя событие второй формы. Фильтр должен применяться к данным первой формы автоматически при изменении текста в поле ввода на второй форме.

Для реализации во второй форме создать:

- событие (можно создать на основе своего делегата или на основе стандартного),

- обработчик события изменения содержимого поля ввода (при изменении должно вызываться созданное событие второй формы).

3.4.4 В первой форме создать подписку на событие, созданное во второй форме. В обработчике события на первой форме выполнить фильтрацию данных.

4 Порядок выполнения работы

4.1 Выполнить все задания из п.3 в одном решении PractWork5. Возможные ошибки требуется обрабатывать. Выполнить форматирование и рефакторинг кода.

4.2 Ответить на контрольные вопросы.

5 Содержание отчета

5.1 Титульный лист

5.2 Цель работы
5.3 Ответы на контрольные вопросы
5.4 Вывод

6 Контрольные вопросы
6.1 Что такое «событие» в C#?
6.2 Как объявить событие на C#?
6.3 Как создать обработчик события?
6.4 Какой класс является родительским для всех классов данных события?
6.5 Какие классы делегатов являются стандартными для создания событий в C#?