

Лабораторная работа №7

События

Цель работы: изучить механизм работы с событиями в языке C#.

Теоретическая часть

Понятие события

Событие представляет собой автоматическое уведомление о том, что произошло некоторое действие.

С точки зрения программы, событием называют сообщение, посланное объектом чтобы проинформировать о совершении некоторого действия.

Класс, содержащий описание события, называется издателем (Publisher). Объект такого класса уведомляет другие объекты, подписавшиеся на событие, о том, что это событие произошло.

Реакция на событие осуществляется с помощью так называемых обработчиков события. Обработчик события – это обычный метод, который выполняет некоторые действия в программе, в случае если состоялось (сгенерировалось) событие.

Подписчик (Subscriber) – это объект, который предоставляет обработчики событий.

События позволяют издателю уведомлять подписчиков о возникновении каких-либо ситуаций.

События действуют по следующему принципу: объект, проявляющий интерес к событию, регистрирует обработчик этого события. Когда же событие происходит, вызываются все зарегистрированные обработчики этого события.

События основаны на делегатах и предоставляют им механизм публикации/подписки. Как и делегаты, события поддерживают групповую

адресацию. Это дает возможность нескольким объектам реагировать на уведомление о событии. Для того, чтобы можно было обрабатывать (запускать) списки обработчиков события, делегат не должен возвращать значение. То есть, делегат может возвращать тип `void`.

События представляют собой специальный вид многоадресного делегата, который можно вызвать только из класса или структуры, в которых он объявлен (класс `Publisher`). Если другие классы или структуры подписываются на событие, их методы обработчиков событий будут вызываться, когда класс `Publisher` будет вызывать событие.

Объявление события

События являются членами класса и объявляются с помощью ключевого слова `event`.

Чаще всего для этой цели используется следующая форма:

`event делегат_события имя_события;`

где `делегат_события` обозначает имя делегата, используемого для поддержки события, а `имя_события` — конкретный объект объявляемого события.

Методы экземпляра и статические методы могут быть использованы в качестве обработчиков событий, но между ними имеется отличие:

- Когда статический метод используется в качестве обработчика, уведомление о событии распространяется на весь класс.
- Когда в качестве обработчика используется метод экземпляра, события адресуются конкретным экземплярам класса. Следовательно, каждый объект определенного класса, которому требуется получить уведомление о событии, должен быть зарегистрирован отдельно.

Порядок работы с событиями

1. Объявить тип делегата в классе.
2. Объявить событие в данном классе или создать другой класс, который содержит объявления события.
3. В некотором методе создать список обработчиков, которые будут вызываться при вызове данного события. Это осуществляется с помощью операторов '=' и '+='. Создание списка означает регистрацию обработчиков для данного события.
4. Вызвать событие (запустить на выполнение) из этого метода.

Примеры работы с событиями:

```
// Объявим тип делегата для обработчика событий
public delegate void MyEventHandler();
8 references
public class MyEvent
{
    ...// Объявляем событие
    ...public event MyEventHandler OnSomeEvent;

    ...// Этот метод вызывается для запуска события
    4 references
    ...public virtual void SomeEvent()
    ...{
    ...    ...// Если событие не пустое
    ...    ...//if (OnSomeEvent != null)
    ...    ...//.....// Вызывается обработчик(и) с помощью делегата
    ...    ...//.....OnSomeEvent();
    ...    ...OnSomeEvent?.Invoke();
    ...}
}
```

Рисунок 1 – Объявление события

```

class Cat
{
    1 reference
    ... public void CatHandler() => Console.WriteLine("Мяу! Событие получил котик");
}

2 references
class Dog
{
    1 reference
    ... public void DogHandler() => Console.WriteLine("Гав! Событие получил пёсик");
}

```

Рисунок 2 – Объявление классов с обработчиками

```

// Обработчик события
1 reference
static void SecondHandler()
{
    ... Console.WriteLine("Событие получил объект класса EventDemo");
}

0 references
public static void SecondDemo()
{
    ... // Создаем объект класса события
    ... MyEvent evt = new MyEvent();

    ... // Создаем котёнка и щенка
    ... Cat kitty = new Cat();
    ... Dog puppy = new Dog();

    ... // Добавим обработчики
    ... evt.OnSomeEvent += SecondHandler;
    ... evt.OnSomeEvent += kitty.CatHandler;
    ... evt.OnSomeEvent += puppy.DogHandler;
    ... // evt.OnSomeEvent -= puppy.DogHandler;

    ... // Запустим событие
    ... // Здесь вызываются все методы, являющиеся обработчиками
    ... evt.SomeEvent();
}

```

Рисунок 3 – Подписка на событие и генерация

Практическая часть

Написать программу для симуляции поляны с грибами.

На поляне изначально находится некоторое количество грибов. Как только на поляну приходит грибник, выводится сообщение: «На поляну пришёл грибник <имя грибника>. В данный момент на поляне N грибов».

Когда грибник собирает X количество грибов, выводится уведомление «Грибник <имя грибника> собрал X грибов».

Когда грибник уходит с поляны, выводится сообщение: «Грибник <имя грибника> ушёл с поляны. На поляне осталось M грибов».

Как только все грибы заканчиваются, все грибники уходят с поляны.

Содержание отчета

1. Титульный лист
2. Цель работы
3. Задание
4. Код программы
5. Результат выполнения