События

№ урока: 12 **Курс:** C# Essential

Средства обучения: Компьютер с установленной Visual Studio

Обзор, цель и назначение урока

Рассмотрение событий.
Рассмотрение паттерна MVP.

Изучив материал данного занятия, учащийся сможет:

- Понимать работу событий.
- Понимать работу паттерна MVP.

Содержание урока

1. События.
2. Паттерн MVP.

Резюме

- **Событийно-ориентированное программирование (event-driven programming)** – парадигма программирования, в которой выполнение программы определяется событиями – действиями пользователя (клавиатура, мышь), сообщениями других программ и потоков, событиями операционной системы (например, поступлением сетевого пакета).
- Событийно-ориентированное программирование, как правило, применяется в трех случаях:
  1. При построении пользовательских интерфейсов (в том числе графических);
  2. При создании серверных приложений в случае, если по тем или иным причинам нежелательно порождение обслуживающих процессов;
  3. При программировании игр, в которых осуществляется управление множеством объектов.
- События позволяют классу или объекту уведомлять другие классы или объекты о возникновении каких-либо ситуаций.
- Класс, отправляющий (или вызывающий) событие, называется издателем.
- Классы, принимающие (или обрабатывающие) событие, называются подписчиками.
- Используйте ключевое слово `event` для объявления события в классе издателя.
- В C# в стандартном приложении Windows Forms или веб-приложении пользователь подписывается на события, вызываемые элементами управления, такими как кнопки и поля со списками. Для просмотра событий, публикуемых элементом управления, и выбора некоторых из них для обработки можно воспользоваться средой IDE Visual C#. IDE автоматически добавит пустой метод обработчика событий и код, необходимый для подписки на событие.
- События имеют следующие свойства:
  1. Издатель определяет момент вызова события, подписчики определяют предпринятое ответное действие.
  2. У события может быть несколько подписчиков. Подписчик может обрабатывать несколько событий от нескольких издателей.
  3. События, не имеющие подписчиков, никогда не возникают.
  4. Обычно события используются для оповещения о действиях пользователя, таких как нажатия кнопок или выбор меню и их пунктов в графическом пользовательском интерфейсе.
  5. Если событие имеет несколько подписчиков, то при его возникновении происходит синхронный вызов обработчиков событий.
  6. В библиотеке классов .NET Framework в основе событий лежит делегат **`EventHandler`** и базовый класс `EventArgs`.

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- Сигнатура обработчика событий должна соответствовать следующим соглашениям:
  1. Метод обработчик события принимает ровно два параметра.
  2. Первый параметр называется `sender` и имеет тип `Object`. Это объект, вызвавший событие.
  3. Второй параметр называется – `e` и имеет тип `EventArgs` или тип производного класса от `EventArgs`. Это данные, специфичные для события.
  4. Тип возвращаемого значения метода обработчика – **`Void`**.
- Чтобы класс мог породить событие, необходимо подготовить три следующих элемента:
  1. Класс, предоставляющий данные для события.
  2. Делегат события.
  3. Класс, порождающий событие.
- **События** – это особый тип многоадресных делегатов, которые можно вызвать только из класса или структуры, в которой они объявлены (класс издателя). Если на событие подписаны другие классы или структуры, их методы обработчиков событий будут вызваны, когда класс издателя инициирует событие.
- События можно пометить как открытые (`public`), закрытые (`private`), защищенные (`protected`), внутренние (`internal`) или `protected internal`.
- Событие можно объявить как статическое событие при помощи ключевого слова `static`. При этом событие становится доступным для вызова в любое время, даже если экземпляр класса отсутствует.
- Событие может быть помечено как виртуальное событие при помощи ключевого слова `virtual`. Это позволяет производным классам переопределять поведение события при помощи ключевого слова `override`.
- События могут быть абстрактными.
- Контекстно-зависимое ключевое слово `add` используется для определения пользовательского метода доступа к событию, вызываемому при подписке клиентского кода к событию. Если указан пользовательский метод доступа `add`, то необходимо также указать метод доступа `remove`.
- Контекстно-зависимое ключевое слово `remove` используется для определения пользовательского метода доступа к событию, вызываемому при отмене подписки клиентского кода от события. Если указан пользовательский метод доступа `remove`, то необходимо также указать метод доступа `add`.

Закрепление материала

- Что такое событие?
- Могут ли события быть статическими?
- Могут ли события быть виртуальными?
- Могут ли события быть абстрактными?
- Могут ли события быть переопределенными?
- Опишите сигнатуру метода обработчика события, согласно соглашению по созданию методов обработчиков событий?
- Ключевое слово `event` используется для объявления события в классе издателя или подписчика?
- Что такое паттерн MVP?

Дополнительное задание

Задание
Измените существующий проект данного урока 003_MVP, расширив его добавлением методов доступа `add` и `remove` к событию.

Самостоятельная деятельность учащегося

Задание 1

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2
Используя Visual Studio, создайте проект по шаблону Console Application.
Используя конструктор диаграмм классов DSL, создайте общую диаграмму классов для паттерна MVP (Model-View-Presenter).

Задание 3
Используя Visual Studio, создайте проект по шаблону WPF Application.
Создайте программу секундомер. Требуется выводить показания секундомера в окне. Окно имеет кнопки запуска, остановки и сброса секундомера. Для реализации секундомера используйте паттерн MVP.

Задание 4
Используя Visual Studio, создайте проект по шаблону WPF Application.
Создайте калькулятор на четыре арифметических действия (сложение, вычитание, умножение и деление). Для реализации калькулятора используйте паттерн MVP.

Задание 5
Зайдите на сайт MSDN.
Используя поисковые механизмы MSDN, найдите самостоятельно описание темы по каждому примеру, который был рассмотрен на уроке, так, как это представлено ниже, в разделе «Рекомендуемые ресурсы», описания данного урока. Сохраните ссылки и дайте им короткое описание.

Рекомендуемые ресурсы

MSDN: Событие (Руководство по программированию на C#)
http://msdn.microsoft.com/ru-ru/library/awbftdfh.aspx

MSDN: Ключевое слово `event` (Справочник по C#)
http://msdn.microsoft.com/ru-ru/library/8627sbea.aspx

MSDN: Подписка и отмена подписки на события (Руководство по программированию на C#)
http://msdn.microsoft.com/ru-ru/library/ms366768.aspx

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com