C# Essential

Події

Після уроку обов'язково

Повторіть цей урок в відео форматі на ITVDN.com

Перевірте як Ви засвоїли даний матеріал на TestProvider.com

Тема

Події

Подієво-орієнтоване програмування

Подієво-орієнтоване програмування (event-driven programming) — парадигма програмування, в якій виконання програми визначається подіями — діями користувача (клавіатура, миша), повідомленнями інших програм і потоків, подіями операційної системи (наприклад, надходженням мережевого пакету).

Події — це особливий тип багатоадресних делегатів, які можна викликати тільки з класу або структури, в якій вони оголошені (клас видавця). Якщо на подію підписані інші класи або структури, їх методи обробників подій будуть викликані, коли клас видавця ініціює подія.

Події дозволяють класу або об'єкту повідомляти інші класи або об'єкти про виникнення будь-яких ситуацій.

Застосування подій

Подієво-орієнтоване програмування, як правило, застосовується в трьох випадках:

- При побудові користувальницьких інтерфейсів (в тому числі графічних);
- При створенні серверних додатків в разі, якщо з тих чи інших причин небажано породження обслуговуючих процесів;
- При програмуванні ігор, в яких здійснюється управління безліччю об'єктів.

Створення подій

Щоб клас міг породити подія, необхідно підготувати три наступних елемента:

- Клас, що надає дані для події.
- Делегат події.
- Клас, який породжує подія.

Створення подій

Для оголошення події в класі видавця, використовується ключове слово `event`

```
// Метод который вызывает событие.
instance.InvokeEvent();
instance.

Console.W                                    g('-', 20));

// Откреп
instance.                                    ntDelegate(Handler2);
instance.

// Delay.
Console.ReadKey();
```



```
// Метод который вызывает событие.
instance.

Console.W                                    g('-', 20));

// Откреп
instance.                                    ntDelegate(Handler2);
instance.

// Delay.
Console.ReadKey();
```



```
public event EventDelegate MyEvent = null;
```

```
public EventDelegate MyEvent = null;
```

На події не можна викликати метод `Invoke()`

Властивості подій

Події мають такі властивості:

- Видавець визначає момент виклику події, підписники визначають відповідну дію.
- У події може бути кілька підписників. Підписник може обробляти кілька подій від декількох видавців.
- Події, що не мають підписників, ніколи не виникають.
- Зазвичай події використовуються для оповіщення про дії користувача, таких як натискання кнопок або вибір меню і їх пунктів у графічному інтерфейсі.
- Якщо подія має кілька підписників, то при його виникненні відбувається синхронний виклик обробників подій.

Сигнатура подій

Сигнатура обробника подій повинна відповідати наступним вимогам:

```
private void button_Click(object sender, EventArgs e)
{
}
```

- Метод обробник події приймає рівно два параметра.
- Перший параметр називається **sender** і має тип object. Це об'єкт, що викликав подія.
- Другий параметр називається - **e** і має тип `EventArgs` або тип похідного класу від `EventArgs`. Це дані, специфічні для події.
- Тип значення, що повертається методу обробника– `void`.

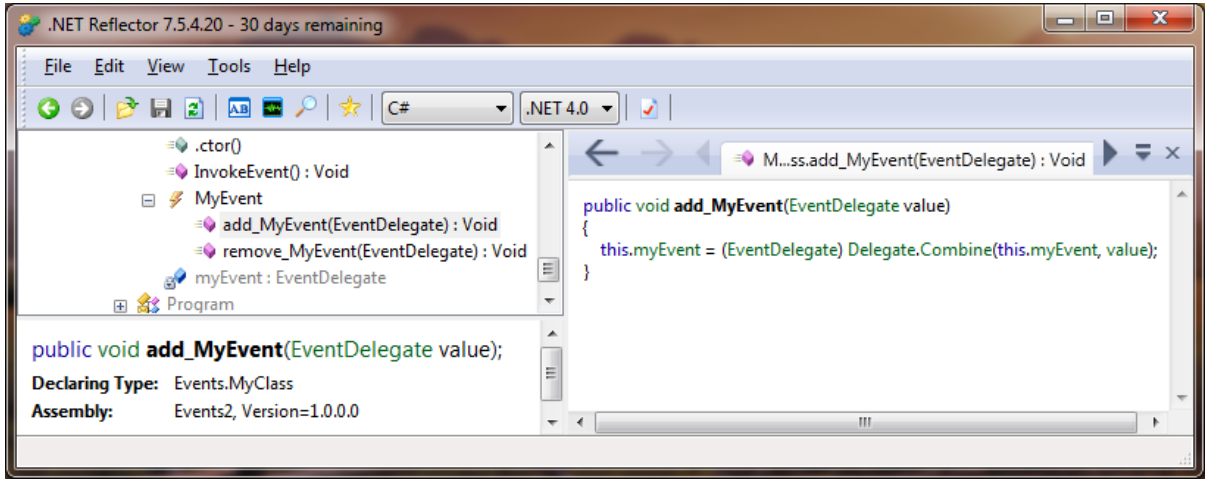
Add-Remove

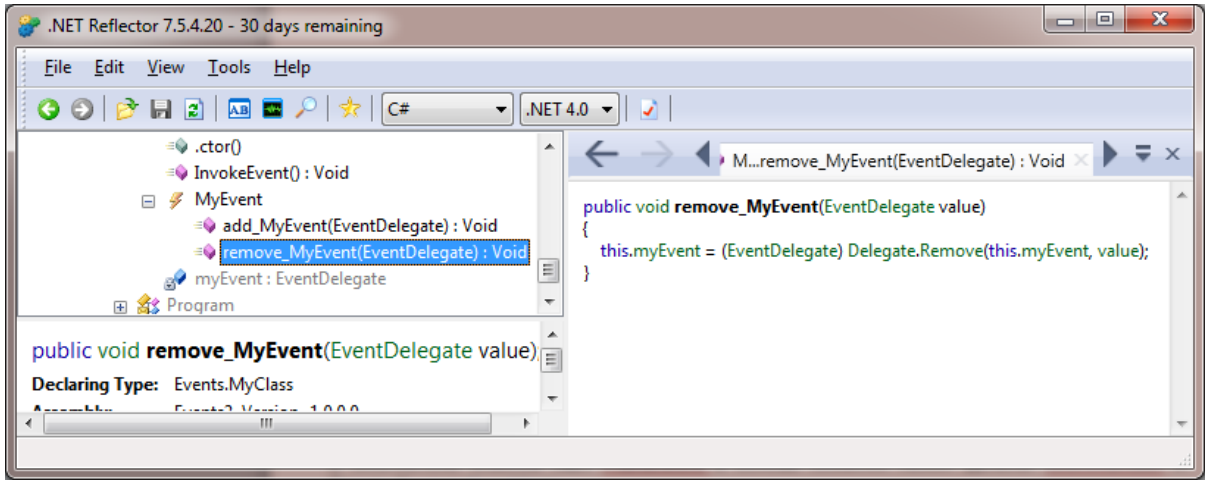
Контекстно-залежне ключове слово add використовується для визначення користувача методу доступу до події, що викликається при підписці клієнтського коду до події. Якщо вказано призначений для користувача метод доступу add, то необхідно також вказати метод доступу remove.

```
public event EventDelegate MyEvent
{
    add { myEvent += value; }
    remove { myEvent -= value; }
}
```

Події

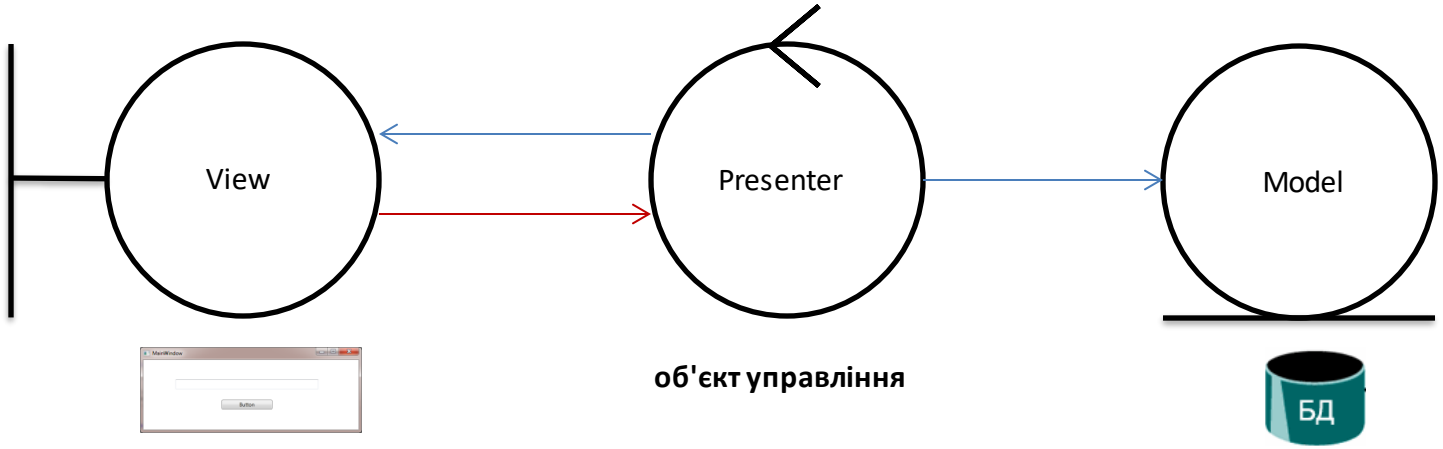
Методи доступу add та remove під .Net Reflector





Model-View-Presenter

MVP



MVP – шаблон проектування призначеного для користувача інтерфейсу, який був розроблений для полегшення автоматичного модульного тестування і поліпшення розподілу відповідальності в презентаційній логіці

Модель (model) являє собою інтерфейс, який визначає дані для відображення або беруть участь в призначеному для користувача інтерфейсі іншим чином

Вид (view) – це інтерфейс, який відображає дані (модель) і маршрутизує призначені для користувача команди (або події) Presenter-у, щоб той діяв над цими даними.

Presenter діє над моделлю і видом. Він витягує дані зі сховища (моделі), і форматує їх для відображення в Вигляді (view).

Q&A

Інформаційний відеосервіс для розробників програмного забезпечення

ITVDN

IT VIDEO DEVELOPERS NETWORK