ВЛАСТИВОСТІ ТА ІНДЕКСАТОРИ

Властивість

Властивість (property) – це спеціальний тип членів класу, завдяки якому спрощується реалізація однієї із основних ідей інкапсуляції даних – дані класу мають бути захищені, а методи доступу до них – відкритими.

```
<модифікатор доступу> <тип> ім'я_властивості
    {
        get {
            // код аксесора читання поля
        }
        set {
            // код аксесора запису поля
        }
    }
```

Приклад властивості

```
class PropertyClass
{
  private int field;  // закрите поле в класі
  public int Field    // властивість для доступу до поля field
  {
      get
      {
          return field;
      }
      set
      {
          if (value > 0)
              field = value;
          else field = 1;
      }
  }
}
```

Приклад використання властивості

```
class  Program
{
    static  void  Main()
    {
        PropertyClass mp = new PropertyClass();
        Console.WriteLine("Введіть поле");
        mp.Field = int.Parse(Console.ReadLine());
        Console.WriteLine("поле = {0}", mp.Field);
        Console.WriteLine("Збільшене поле:{0}", ++mp.Field);
    }
}
```

Приклад властивості для читання

```
class MyPropertyClass {
      private string name = "MyName";
      public string Name  // Властивість лише для читання
      {
            get  {    return name;   }
      }
}
class Program
{
      static void Main()
      {
         MyPropertyClass mp = new MyPropertyClass();
         Console.WriteLine("name = " + mp.Name);
         mp.Name = "NewName";
      // тут буде помилка - властивість лише для читання !
        }
}
```

Авто-реалізовані властивості

```
class Cat
{
    // Auto-implemented properties.
    public int Age { get; set; }
    public string Name { get; set; }
}
Cat cat = new Cat { Age = 10, Name = "Fluffy" };
-----------------------------------
List<Cat> cats = new List<Cat>
{
    new Cat(){ Name = "Sylvester", Age=8 },
    new Cat(){ Name = "Kitty", Age=2 },
    new Cat(){ Name = "Tom", Age=14 }
};
```

Приклад властивості

```
struct TwoDPoint
    {
   //Read/write auto-implemented properties.
        public int X { get; private set; }
        public int Y { get; private set; }

        public TwoDPoint(int x, int y)
        {
            X = x;
            Y = x;
        }
```

Особливості властивостей

Поле розміщується у пам'яті, а властивість – ні.

Властивість є логічним полем, доступ до якого здійснюється через аксесори get та set.

Властивості, як і поля, мають модифікатор доступу.

Властивості, як і поля, можуть бути статичними.

Властивість, на відміну від методу, не може перевантажуватись.

Формальна змінна value є видимою та може бути використана лише в межах властивості, – звертання до змінної поза властивістю призведе до синтаксичної помилки.

Індексатор

```
<модиф. доступу> <тип> this [<тип індексу> <змінна індексу>]
{
        // Аксессор для отримання даних
        get
        {
                // Повернення значення, яке задається індексом
        }
        // Аксессор для встановлення даних
        set
        {
                //Встановлення значення, яке задається індексом
        }
}
```

Використання індексатора

```
class  IndexClass
 {
      private int size;                    // розмір масиву
      private int[] arr;                   // оголошення масиву
      public MyIndexClass(int sizeAr)  // конструктор
      {
        size = sizeAr;
        arr = new int[size];               // створення масиву
      }
      public int this [int ind]            // це індексатор
      {
        get //отримуємо елемент масиву за зверненням до індексатора
        {
                if ((ind >= 0) && (ind < size)) {     return arr[ind];     }
                return 0;
        }
        set // присвоюємо елемент масиву через індексатор
        {
                if ((ind >= 0) && (ind < size)) { arr[ind] = value; }
        }
      }
}
```

Використання індексатора

```
class Program
{
        static void Main()
        {
           IndexClass mic = new IndexClass(5);
           for (int i = 0; i < 5; i++)
           {
                // виклик аксесора set індексатора
                mic[i] = i;

                // виклик аксесора get індексатора
                Console.WriteLine("mic [{0}] = {1}", i, mic[i]);
           }
        }
}
```

Індексатор з дійсним індексом

```
public int this [double idx] {
      get {
        int index;
        // заокруглення до найближчого цілого int-значення
        if ( (idx - (int) idx) < 0.5 )
                index = (int) idx;
        else
                index = (int) idx + 1;
        if ( (index >= 0) && (index < size) )
                {     return arr[index];     }
        return 0;
      }
      set {
        int index;
        if ( (idx - (int) idx) < 0.5 )
                index = (int) idx;
        else
                index = (int) idx + 1;
        if ( (ind >= 0) && (ind < size) ) { arr[ind] = value; }      }
    }
```

Використання індексатора

Індексатор може не використовувати базовий масив.

```
class PwrOfTwo {
// Доступ до логічного масиву, який містить степені числа 2
  public int this [int index] { //Обчислюємо і повертаємо ступінь числа 2
    get {
      if ( (index >= 0) && (index < 16) )
        return pwr (index);
      else return -1;
    }
    // set-аксесор відсутній
  }
  int pwr (int p) {
    int result = 1;
    for (int i = 0; i <p; i ++)
      result *= 2;
    return result;
  }
}
```

Використання індексатора

```
class UsePwrOfTwo {
  public static void Main () {
    PwrOfTwo pwr = new PwrOfTwo ();
    Console.Write ("Перші 8 степенів числа 2:");
    for (int i = 0; i < 8; i ++)
      Console.Write (pwr [i] + "");
    Console.WriteLine ();

    Console.Write ("Виведення помилок:");
    Console.Write (pwr [-1] + "" + pwr [17]);
  }
}
```

Особливості індексаторів

На використання індексаторів накладається два обмеження.

По-перше, оскільки в індексаторі не визначається область пам'яті, одержуване індексатором значення не можна передавати методу в якості ref- або out-параметра.

По-друге, індексатор повинен бути членом екземпляра свого класу, тому його не можна оголошувати з використанням ключового слова static.

Властивості та індексатори

1. Доступ до властивості здійснюється за її ідентифікатором, індексатор не має власного ідентифікатору, доступ здійснюється за індексом елементу.

2. Аксесор get властивості не має параметрів, в той час як get індексатора має один (або більше) параметрів – *індекс*.

3. Аксесор set властивості має неявний параметр value, а set індексатора крім value має ті самі індекси, що і його get.

4. Властивість може бути статичним членом класу, індексатор – ніколи, оскільки він визначається через посилання this.

5. Властивості не перевантажуються в той час, як індексатор може бути перевантажений за рахунок використання індексу іншого типу, або іншої кількості індексів.