Вопрос:

Что такое индексатор и как его создать в языке C#?

Ответ:

Особенным случаем перегрузки операторов является перегрузка оператора «квадратные скобки». В языке C# для такой перегрузки используется отдельная конструкция языка, называющаяся индексатором. Индексатор напоминает свойство (property), однако у данного свойства предусмотрены параметры, которые указываются в квадратных скобках. Пример объявления индексатора:

```
public int this[int i] {
	get
	{
		return this.Num + i;
	}
	set
	{
		this.Num = value + i;
	}
}
```

В этом случае вместо имени свойства указывается ключевое слово this, а после него в квадратных скобках даются параметры. Параметров может быть произвольное количество, причем их использование никак не ограничено. Однако большинство программистов интуитивно воспринимают квадратные скобки как оператор для доступа к массиву или коллекции. Поэтому в качестве параметра обычно передают какой-либо ключ в числовой или символьной форме или набор ключей в случае много мерного массива. Почему в языке C# не используется оператор перегрузки квадратных скобок в обычной форме? Потому что форма индексатора позволяет перегрузить сразу два оператора чтения и записи, что является наиболее привычным для программиста поведением при перегрузке квадратных скобок.

Индексаторы позволяют индексировать экземпляры класса или структуры точно так же, как и массивы. Индексированное значение можно задавать или получать без явного указания типа или экземпляра элемента. Индексаторы действуют как свойства, за исключением того, что их акцессоры принимают параметры.

Большинство программистов на C# сначала используют индексатор при работе с массивом. Массив используется для хранения ряда похожих,

связанных переменных под одним и тем же именем, причем каждая переменная доступна с помощью индексного номера, заключенного в квадратные скобки. Например:

```
thirdItem = items[3];
```

Часто бывает полезно включить обозначение квадратных скобок для новых классов. Это может быть связано с тем, что класс используется для хранения связанной информации аналогично массиву, или просто потому, что номер индекса может быть полезен при вычислении или поиске. Добавление индексатора в класс обеспечивает эту функциональность.

Самый простой вариант индексатора - одномерный тип. Одномерный индексатор принимает одно значение между квадратными скобками при использовании. Стандартный синтаксис, используемый для объявления индексатора, аналогичен синтаксису, используемому для определения методов доступа get и set свойства. Однако вместо определения имени свойства методы доступа объявляются для этого [] следующим образом:

```
public data-type this[index-type index-name]

{

   get {}

   set {}

}
```

В определении синтаксиса тип данных определяет тип информации, которая будет возвращена при запросе индексатора, и тип, который будет необходим при установке значения. Тип индекса указывает тип данных самого индексатора. Это позволяет объявлять индексаторы, которые не основаны на целочисленных значениях, что позволяет использовать аналогичную функциональность, например, для хэш-таблицы. Имя индекса - это переменная, содержащая значение индекса, которое может быть использовано при обработке методов доступа get и set.

Метод доступа get необходим для индексатора и должен возвращать значение типа data-type. Метод доступа set определен для индексаторов с возможностью записи и опущен, если требуется вариант только для чтения.