



## Свойства

Последнее обновление: 30.10.2015



В VB.NET кроме обычных методов и конструкторов также есть специальные методы доступа - **свойства**. Свойства помогают получить доступ к полю класса, узнать его значение или наоборот установить его. Для объявления свойства используется ключевое слово **Property**:

```
1 Public Class State
2
3     Dim name_ As String
4
5     Public Property Name As String
6         Get
7             Return name_
8         End Get
9         Set(value As String)
10            name_ = value
11        End Set
12    End Property
13
14 End Class
```

У нас определено поле **name\_**, оно закрытое, и чтобы получить к нему доступ из других частей программы и классов, мы будем использовать свойство **Name**. При объявлении мы также задаем тип поля, для которого это свойство создано (в данном случае String). Обратите внимание на синтаксис свойства: оно имеет два блока - **Get** и **Set**. В блоке **Get**:

```
1 Get
2     Return name_
3 End Get
```

мы получаем с помощью оператора **Return** значение переменной **name\_**, оно же и будет значением данного свойства. В блоке **Set** мы осуществляем присваивание значения переменной **name\_**:

```
1 Set(value As String)
2     name_ = value
```

3 End Set

Параметр value представляет передаваемое значение. Мы можем использовать данное свойство следующим образом:

```
1 Sub Main()  
2  
3     Dim s As State = New State()  
4  
5     'Устанавливаем свойство - отрабатывает блок Set  
6     s.Name = "Russia"  
7  
8     'Получаем значение свойства и присваиваем его переменной - отрабатывает  
9     Dim stateName = s.Name  
10  
11     Console.ReadLine()  
12 End Sub
```

Несмотря на то, что для свойства определен модификатор Public, мы можем задать модификатор и для блока Set или Get:

```
1 Public Property Name As String  
2     Private Get  
3         Return name_  
4     End Get  
5     Set(value As String)  
6         name_ = value  
7     End Set  
8 End Property
```

В этом случае мы уже не сможем в модуле получить значение свойства:

```
1 Sub Main()  
2  
3     Dim s As State = New State()  
4  
5     'Устанавливаем свойство - отрабатывает блок Set  
6     s.Name = "Russia"  
7  
8     'Получаем значение свойства - блок Get объявлен как Private,  
9     'поэтому здесь будет ошибка  
10    Dim stateName = s.Name  
11  
12    Console.ReadLine()  
13 End Sub
```

## Read-Only и Write-Only в определениях свойств

Иногда необходимо сделать свойства доступными только для чтения или записи. В этом случае мы можем использовать ключевые слова **ReadOnly** и **WriteOnly**. Если мы хотим сделать свойство только для чтения, то надо использовать слово **ReadOnly** и опустить Set. Если надо создать свойство только для записи, то используется слово **WriteOnly** и опускается Get:

```
1 Public Class State
2
3     Dim name_ As String
4
5     Dim population_ As Integer
6
7     ReadOnly Property Name As String
8         Get
9             Return name_
10        End Get
11    End Property
12
13    WriteOnly Property Population As Integer
14        Set(value As Integer)
15            population_ = value
16        End Set
17    End Property
18
19 End Class
```

### Автоматические свойства

В .NET 4 была добавлена новая функциональность - **автоматические свойства**.

Благодаря таким свойствам можно использовать сокращенный синтаксис объявления свойств и не создавать дополнительно поля в классе, с которыми связаны эти свойства:

```
1 Public Class State
2
3     Public Property President() As String
4     Public Property Area() As Double
5
6 End Class
```

Правда, использование автосвойств имеет ограниченные возможности, во-первых, поскольку мы не можем при получении или присвоении значения добавить в свойства дополнительную логику; во-вторых, мы не можем таким образом создавать свойства только для записи или только для чтения.

[Назад](#) [Содержание](#) [Вперед](#)



ТАКЖЕ НА METANIT.COM

|   |   |  |  |
|---|---|--|--|
| <b>Ассемблер MASM.<br/>Установка и начало ...</b><br><br>3 месяца назад · 4 коммент...<br>Ассемблер MASM.<br>Установка и начало<br>работы, Visual Studio, ... | <b>Клиент на Xamarin<br/>Forms для SignalR</b><br><br>10 дней назад · 1 коммента...<br>Клиентское приложение<br>на Xamarin Forms для<br>SignalR в ASP.NET Core, ... | <b>Параметры строки<br/>запроса</b><br><br>5 месяцев назад · 1 коммен...<br>Параметры строки<br>запроса query string в<br>приложении Blazor на ... | <b>Подк</b><br><br>5 меся<br>Библи<br>подкл<br>даннь |
|---|---|--|--|

1 Комментарий

1 Войти ▾

G

Присоединиться к обсуждению...

войти с помощью

или через DISQUS ?

Имя



Поделиться

Лучшие Новые Старые

A

Alexandr

7 лет назад

Теперь можно [https://msdn.microsoft.com/...](https://msdn.microsoft.com/) писать автоматические свойства с ReadOnly (начиная с 2015 студии, как я понял msdn). Однако с WriteOnly по-прежнему недоступно.

```
Public ReadOnly Property Tags As New List(Of String)
Public ReadOnly Property Name As String = ""
Public ReadOnly Property File As String
```

1 0 Ответить • Поделиться ›

Помощь сайту

YooMoney:  
410011174743222

Qiwi:

[qiwi.com/n/METANIT](https://qiwi.com/n/METANIT)

Перевод на карту

**Номер карты:**

4048415020898850

[Вконтакте](#) | [Телеграм](#) | [Twitter](#) | [Помощь сайту](#)

Контакты для связи: metanit22@mail.ru

Copyright © metanit.com, 2023. Все права защищены.