

Форматиране на код

Да форматираме програмния код правилно?



Учителски екип

Обучение за ИТ кариера

<https://it-kariera.mon.bg/e-learning/>



Съдържание

1. Защо трябва да форматираме кода?
2. Форматиране на методи
3. Форматиране на типове
4. Често срещани грешки
5. Подравняване
6. Автоматизирани инструменти



Защо трябва да форматираме кода?

```
public    const    string    FILE_NAME
="example.bin" ; static void Main    ){
FileStream    fs=    new FileStream(FILE_NAME, FileMode
.    CreateNew)    // Create the writer    for data .
;BinaryWriter w=new    BinaryWriter    (    fs    );
// Write data to    Test.data.
for(    int i=0;i<11;i++){w.Write((int)i);}w    .Close();
fs    .    Close    (    ) // Create the reader    for data.
;fs=new FileStream(FILE_NAME,FileMode.    Open
,    FileAccess.Read)    ;BinaryReader    r
= new BinaryReader(fs);    // Read data from    Test.data.
    for (int i = 0; i < 11; i++){ Console    .WriteLine
(r.ReadInt32    ())
; }r    .    Close    (    ); fs .    Close    (    ); }
```

Основни правила за форматиране на кода

- Цели на доброто форматиране
 - Да подобри четливостта на кода
 - Да подобри възможностите за поддръжка на кода
- Основен принцип на форматирането на кода:

Форматирането на програмния код трябва да показва неговата логическа структура.

- Всеки стил на форматиране, следващ този принцип, е добър
- Всеки друг не е

Форматиране на блокове

- Слагайте { и } **сами** на ред под съответния обграждащ блок
- Вмъкнете навътре съдържанието на блока с един [Tab]
 - Visual Studio ще замени този [Tab] с 4 паузи
- Пример:

```
if (some condition)
{
    // Block contents indented by a single [Tab]
    // VS will replace the [Tab] with 4 spaces
}
```

Празни редове между методите

- Използвайте празен ред, за да разделите методите:

```
public class Factorial
{
    private static ulong CalcFactorial(uint num)
    {
        if (num == 0)
            return 1;
        else
            return num * CalcFactorial(num - 1);
    }

    static void Main()
    {
        ulong factorial = CalcFactorial(5);
        Console.WriteLine(factorial);
    }
}
```

Винаги използвайте { и } след if
(тук за това няма място)

Оставете празен ред
между методите

Вмъкване на методи навътре

- Методите трябва да са вмъкнати с един [Tab] навътре от тялото на класа
- Тялото на метода също да е вмъкнато навътре с един [Tab]

```
public class IndentationExample
{
    private int Zero()
    {
        return 0;
    }
}
```

Целият метод е вмъкнат навътре с
един [Tab]

Тялото на метода също е вмъкнато

Скоби в декларирането на методи

- Скобите в декларирането на методи трябва да са форматиранни така:

```
private static ulong CalculateFactorial(uint num)
```



- Не поставяйте паузи между скобите:

```
private static ulong CalculateFactorial ( uint num )
```

```
private static ulong CalculateFactorial (uint num)
```



- Същото се отнася за условия с **if** и цикли **for**:

```
if (condition) { ... }
```

Скобите трябва да са на отделен ред

Отделяне на параметри

- Отделяйте параметрите на метод със запетая и после пауза
 - Не слагайте паузата преди запетаята
 - Примери:

```
private void RegisterUser(string username, string password)
```

```
RegisterUser("nakov", "s3cr3t!p@ssw0rd");
```



- Лоши примери:

```
private void RegisterUser(string username,string password)
```

```
private void RegisterUser(string username ,string password)
```

```
private void RegisterUser(string username , string password)
```



Празни редове в тялото на метода

- Използвайте празен ред, за да отделите логически цялостни части:

```
private List<Report> PrepareReports()
{
    List<Report> reports = new List<Report>();

    // Create incomes reports
    Report incomesSalesReport = PrepareIncomesSalesReport();
    reports.Add(incomesSalesReport);
    Report incomesSupportReport = PrepareIncomesSupportReport();
    reports.Add(incomesSupportReport);

    // Create expenses reports
    Report expensesPayrollReport = PrepareExpensesPayrollReport();
    reports.Add(expensesPayrollReport);
    Report expensesMarketingReport = PrepareExpensesMarketingReport();
    reports.Add(expensesMarketingReport);

    return reports;
}
```

Празен ред

Празен ред

Празен ред

Форматиране на типове

- Форматиране на класове – **ВМЪКНЕТЕ** тялото на класа с **един**[Tab]

```
public class Dog
{
    // Static variables
    public const string Species =
        "Canis Lupus Familiaris";
    // Instance variables
    private int age;
    // Constructors
    public Dog(string name, int age)
    {
        this.Name = name;
        this.age = age;
    }
}
```

```
// Properties
public string Name { get; set; }
// Methods
public void Breathe()
{
    // TODO: breathing process
}
public void Bark()
{
    Console.WriteLine("wuf-wuf");
}
}
```

Форматиране на условни команди и цикли

- Винаги използвайте { } за вложените команди, дори когато е само една команда
- Винаги оставяйте празен ред след { } блока!
- Вмъкнете навътре командите в тялото на блока
- Винаги поставяйте { на нов ред
- Не вмъквайте с повече от един [Tab]

Форматиране на условни команди и цикли

■ Пример:

```
for (int i = 0; i < 10; i++)  
{  
    Console.WriteLine("i = {0}", i);  
}
```

Липсват скобите
{ и }

■ Лоши примери:

```
for (int i=0; i<10; i++)  
    Console.WriteLine("i={0}", i);
```

Никога не слагайте
много команди на един
и същи ред!

```
for (int i=0; i<10; i++) Console.WriteLine("i={0}", i);
```

```
for (int i=0; i<10; i++) {  
    Console.WriteLine("i={0}", i);  
}
```

Тази скоба { трябва да е
на следващия ред

Използване на празни редове

- Логически разделят несвързани части от програмния код

```
public static void PrintList(List<int> ints)
{
    Console.Write("{ ");
    foreach (int item in ints)
    {
        Console.Write(item);
        Console.Write(" ");
    }

    Console.WriteLine("}");
}

static void Main()
{
    // ...
}
```

Празен ред след
блока foreach

Празен ред
отделя
методите

- Не слагайте празни редове без нужда!

Грешно поставени празни редове – Пример

```
public static void PrintList(List<int> ints)
{
    Console.Write("{ ");
    foreach (int item in ints)
    {
        Console.Write(item);

        Console.Write(" ");

    }
    Console.WriteLine("}");
}
static void Main()
{
    // ...
}
```

За какво служат тези празни редове?

Прекъсване на дълги редове

- Преминете на нов ред след препинателния знак
- Вмъкнете навътре втория ред с един [Tab]
- Не вмъквайте следващите редове повече
- Примери:

```
if (matrix[x, y] == 0 || matrix[x - 1, y] == 0 ||  
    matrix[x + 1, y] == 0 || matrix[x, y - 1] == 0 ||  
    matrix[x, y + 1] == 0)  
{ ...
```

```
DictionaryEntry<K, V> newEntry =  
    new DictionaryEntry<K, V>(  
        oldEntry.Key,  
        oldEntry.Value);
```

Неправилни начини да се прекъсне дълъг ред

```
if (matrix[x, y] == 0 || matrix[x-1, y] ==  
    0 || matrix[x+1, y] == 0 || matrix[x,  
    y-1] == 0 || matrix[x, y+1] == 0)  
{ ...
```

```
if (matrix[x, y] == 0 || matrix[x-1, y] == 0 ||  
    matrix[x+1, y] == 0 || matrix[x, y-1] == 0 ||  
    matrix[x, y+1] == 0)  
{ ...
```

```
DictionaryEntry<K, V> newEntry  
= new DictionaryEntry<K, V>(oldEntry  
    .Key, oldEntry.Value);
```

Подравняване

- Всички видове подравняване се смятат за вредоносни
 - Подравняването е трудно за поддръжка!
- Лоши примери:

```
int          count      = 0;  
DateTime     date       = DateTime.Now.Date;  
Student      student    = new Student();  
List<Student> students  = new List<Student>();
```

```
matrix[x, y]          = 0;  
matrix[x + 1, y + 1]  = 0;  
matrix[2 * x + y, 2 * y + x] = 0;  
matrix[x * y, x * y]  = 0;
```

Представете си да трябва да преименувате Student на SchoolStudent

Автоматизирани инструменти

- Възползвайте се от своята IDE при форматирането на кода
- [Ctrl] + K + D във Visual Studio
 - Автоматично подравняване
- Анализиране на стила на кода
 - StyleCop
 - <https://stylecop.codeplex.com/>
 - JetBrains ReSharper
 - <https://www.jetbrains.com/resharper/>

Обобщение

1. Съвети за форматиране

- Уверете се, че форматирането показва целта на кода

2. Конвенции при форматирането

- Блокове, типове, параметри на методи
- Логическо отделяне на свързани блокове текст

3. Автоматизиран анализ на кода и инструменти за преработка



Форматиране на код



Въпроси?

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"



Министерство
на образованието
и науката



Национална
програма
„Обучение за
ИТ кариера“

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**



SoftUni
Foundation

