## Стандарт оформления кода

1. Интервалы:
* Вставлять пробелы в списке аргументов при объявлении методов.

*int Method( int input ) {*

 *return input;*

*}*

* Вставлять пробелы в списке аргументов при вызове методов.

*int Method() {*

 *Console.WriteLine( "In Method" );*

 *return 3;*

*}*

* Вставлять пробелы после зарезервированных слов и открывающейся скобкой.

*int Method() {*

 *if ( a > b )*

 *return 0;*

 *return 3;*

*}*

* Вставлять пробелы между круглыми скобками в выражениях.

*double Function( double a, double b ) {*

 *return a \* ( b - a );*

*}*

* Вставлять пробелы внутри круглых скобок для управляющих операторов.

*int Method() {*

 *if ( a > b )*

 *return 0;*

 *return 3;*

*}*

* Вставлять пробел после операции приведения типа.

*ArrayList names = new ArrayList();*

*names.Add("John");*

*string name = (string) names[0];*

* Вставлять пробел после и перед двоеточием при описания наследования.

*class MyClass : IDisposable*

*{*

 *// ...*

*}*

* Вставлять пробелы после запятых.

*int Sum( int a, int b ) {*

 *return a + b;*

*}*

* Вставлять пробел после ‘;’ в for.

*int Method() {*

 *for ( int i = 0; i < 10; ++i ) {*

 *OtherMethod();*

 *}*

*}*

* Вставлять пробел до и после бинарных операций.

*void Method() {*

 *int result = 1 + 2 \* 3;*

*}*

1. Новые строки:
* Располагать открывающуюся фигурную скобку с новой строки при описании типов.

*class MyClass*

*{*

 *// ...*

*}*

* Располагать открывающуюся фигурную скобку с новой строки при описании анонимных методов.

*timer.Tick += delegate (object sender, EventArgs e)*

*{*

 *MessageBox.Show(this, "Timer ticked");*

*};*

* Располагать ключевое слово catch с новой строки.

*try*

*{*

 *// ...*

*}*

*catch ( Exception e )*

*{*

 *// ...*

*}*

* Размещать стрелки в инициализаторах с новой строки.

*void Method() {*

 *MyObject obj = new MyObject {*

 *X1 = 0,*

 *Y1 = 1,*

 *X2 = 2,*

 *Y2 = 3*

 *};*

*}*

* Располагать члены анонимных типов с новой строки.

*void Method() {*

 *var obj = new {*

 *X1 = 0,*

 *Y1 = 1,*

 *X2 = 2,*

 *Y2 = 3*

 *};*

*}*

* Располагать выражение LINQ запроса с новой строки.

void Method() {

 var q = from c in cust

 from o in c.Orders

 orderby o.Total descending

 select new { c.Name, c.OrderID };

}

1. Отступы:
* Отступ перед блоком кода.

*class MyClass*

*{*

 *int Method() {*

 *return 3;*

 *}*

*}*

* Отступ в конструкции case.

*switch ( name ) {*

 *case "John":*

 *break;*

*}*

1. Перенос по словам:
* Допускается оставлять блок в одну строчку.

*public int Age { get { return age; } }*

* Допускается оставлять инструкции и объявление однотипных членов на одной строке.

*int age1 = 18; string name1 = "John"; string sername1 = "Ball";*

*int age1 = 20; string name2 = "Bill"; string sername2 = "Smith";*

1. Программные имена:
* Имя класса с большой буквы «холмами».

*class MyClass*

*{*

 *//----*

*}*

* Имя свойства класса с большой буквы «холмами».

*class MyClass*

*{*

 *public string MyClassText { get; set; }*

*}*

* Имя метода класса с маленькой буквы «холмами».

*class MyClass*

*{*

 *public string getMyClassText() {*

 *}*

*}*

* Имя поля с маленькими буквами через символ ‘\_’.

*class MyClass*

*{*

 *private string my\_class\_text;*

*}*

1. Требования к структуре кода:
* Каждый структурный оператор языка предусматривающий ‘{’ и ‘}’ должен их содержать.

*If ( a > b ) {*

 *++a;*

*}*

* Код класса должен быть разделен на логически блоки с помощью #region - #endregion.

 *public class MoveThumb : Thumb*

 *{*

 *#region Конструкторы*

 *public MoveThumb();*

 *#endregion*

 *#region События*

 *private void MoveThumb\_DragDelta( object sender, DragDeltaEventArgs e );*

 *#endregion*

 *}*