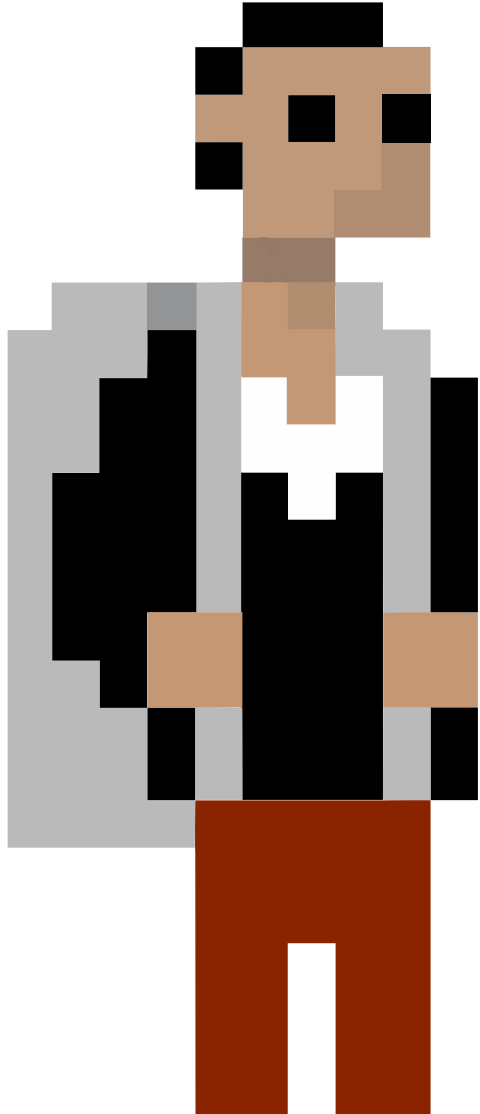
A man with a beard and glasses, wearing a blue shirt, is seated in the background, slightly out of focus. In the foreground, there is a large number of brown stuffed monkeys, with one in sharp focus in the center. The scene is indoors, with a window and some foliage visible in the background.

Motz Codes Live

@JamesMontemagno
motzcod.es

C# 6: Clean Up Your Code!
October 9th, 2015

Who is Motz?



James
Montemagno
Developer Evangelist, Xamarin

james@xamarin.com

motzcod.es

[@JamesMontemagno](https://twitter.com/JamesMontemagno)

C# 6.0 => Clean up your code

July 2015

- Auto-property Enhancements
- Expression-Bodied Members
- Null Propagator
- String Interpolation
- MORE!
 - Await in catch/finally
 - Exception Filters
 - nameof Operator
 - Dictionary_INITIALIZER
 - Import Static Type

.NET 4.6

VS 2015

Xamarin Studio

C# 6.0 => Monkeys

```
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; private set; }
    public string Name { get; set; }
    public string Location { get; set; }

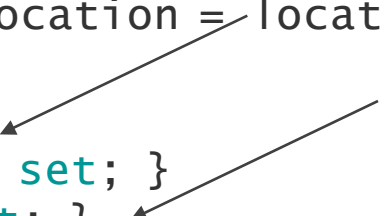
    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? DayOfWeek.Friday : DayOfWeek.Monday; }
    }
    public void Print()
    {
        Console.WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

Auto properties

```
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; private set; }
    public string Name { get; set; }
    public string Location { get; set; }

    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? DayOfWeek.Friday : DayOfWeek.Monday; }
    }
    public void Print()
    {
        Console.WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

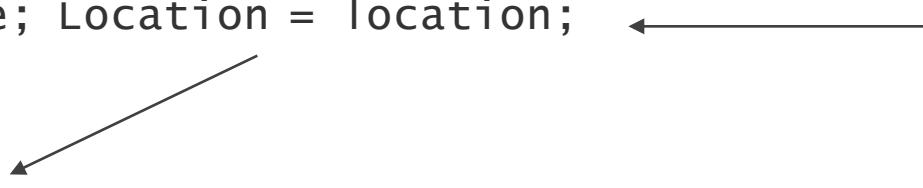


Auto properties

```
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; }
    public string Name { get; }
    public string Location { get; set; }

    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? DayOfWeek.Friday : DayOfWeek.Monday; }
    }
    public void Print()
    {
        Console.WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

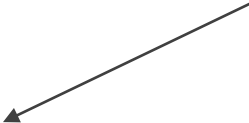


Initializers

```
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? DayOfWeek.Friday : DayOfWeek.Monday; }
    }
    public void Print()
    {
        Console.WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```



Using static

```
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? DayOfWeek.Friday : DayOfWeek.Monday; }
    }
    public void Print()
    {
        Console.WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```


Using static

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay
    {
        get { return Id == 0 ? Friday : Monday; }
    }
    public void Print()
    {
        WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```


Expression-bodied functions!

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print()
    {
        WriteLine("We printed some stuff");
    }
    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

Expression-bodied functions!

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

String Interpolation

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName
    {
        get
        {
            return string.Format("Monkey {0} lives in {1} with Id of {2}",
                                  Name, Location, Id);
        }
    }
}
```

String Interpolation

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName
    {
        get
        {
            return $"Monkey {Name} lives in {Location} with Id of {Id}";
        }
    }
}
```

String Interpolation

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => return Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName
    {
        get
        {
            return $"Monkey {Name} lives in {Location} with Id of " +
                $"{(Id == 0 ? \"Bananas\" : Id)}";
        }
    }
}
```


String Interpolation

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName =>
        $"Monkey {Name} lives in {Location} with Id of " +
        $"{(Id == 0 ? \"Bananas\" : Id)}";
}
```

C# 6.0 => Monkeys

```
using static System.Console;
using static System.DayOfWeek;
public class Monkey
{
    public Monkey() { }
    public Monkey(int id, string name, string location)
    {
        Id = id; Name = name; Location = location;
    }

    public int Id { get; } = 0;
    public string Name { get; } = "Sofia";
    public string Location { get; set; } = "South America";

    public DayOfWeek MostActiveDay => Id == 0 ? Friday : Monday;

    public void Print() => WriteLine("We printed some stuff");

    public string DisplayName =>
        $"Monkey {Name} lives in {Location} with Id of " +
        $"{(Id == 0 ? \"Bananas\" : Id)}";
}
```

?



Null-Conditional Operators

```
var length = 0;  
if(customers != null)  
    length = customers.Length;
```

```
int? length = customers?.Length;  
// null if customers is null
```

```
int length = customers?.Length ?? 0;
```

```
Customer first = null  
if(length > 0)  
    first = customers[0];
```

```
Customer first = customers?[0];  
// null if customers is null
```

If null then null, if not then dot

Some more cool stuff

Await in catch & finally blocks

```
Resource res = null;  
try  
{  
    res = await Resource.OpenAsync(...); // You could do this. ...  
}  
catch(ResourceException e)  
{  
    await Resource.LogAsync(res, e); // Now you can do this ...  
}  
finally  
{  
    if (res != null)  
        await res.CloseAsync(); // ... and this.  
}
```

nameof expressions

```
public static void GetMonkey(int count)
{
    if(count < 0)
        throw new ArgumentNullException("count");

    var monkeys = GetMonkeys();
    WriteLine("Name: " + monkeys[0].Name);
    // prints "Name Sofia"
}
```

nameof expressions


```
public static void GetMonkey(int count)
{
    if(count < 0)
        throw new ArgumentNullException(nameof(count));

    var monkeys = GetMonkeys();
    WriteLine(nameof(monkeys[0].Name) + ": " + monkeys[0].Name);
    // prints "Name Sofia"
}
```



Resources


- Awesome Microsoft Docs
 - <http://bit.ly/csharp6-features>
- C# 6 Video with Mads:
 - <https://channel9.msdn.com/Series/ConnectOn-Demand/211>
- Motz Codes Live:
 - <https://www.youtube.com/user/jamesmontemagno>
 - <https://github.com/jamesmontemagno/MotzCodesLive>

Pinned Tweet

 **James Montemagno** @JamesMontemagno · May 1

C# 6 in 8 minutes!
channel9.msdn.com/Series/Connect...
Simply fantastic.

 **Microsoft Channel 9**



What's new in C# 6




Mads Torgersen
Language PM for C#

7 minutes, 49 seconds

What's new in C# 6.0 (Channel 9)

C# 6 adds a lot of small but useful language features to remove boilerplate and clean up your code. This video takes you on a whirlwind tour through all the new language constructs.

[View on web](#)

  252  384 