MATHEMA

Dynamisch und Gefährlich?

C# Dynamics in freier Wildbahn

19.07.2013 V0.1

MATHEMA

Tim Bourguignon

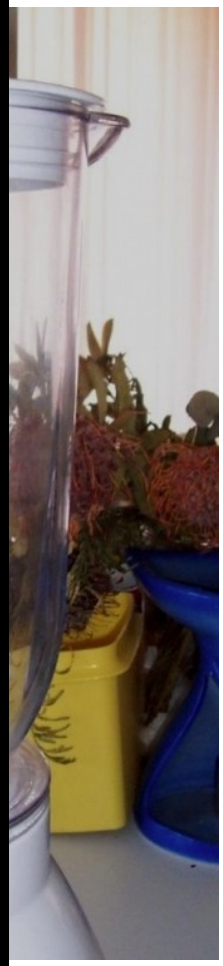
Senior Entwicker

Dynamics? Nooooooooo....

Dynamics? Noooooooo....

Dynamics? Noooooooo....

Dynamics? Noooooooo....



Compiler says *meep*


```
string lang = "C#";
lang++;

int theAnswer = 42;
theAnswer.ToUpper();
```

Compiler says *meep*


```
string lang = "C#";
lang++;

int theAnswer = 42;
theAnswer.ToUpper();
```

✖	1	Operator '++' cannot be applied to operand of type 'string'
✖	2	'int' does not contain a definition for 'ToUpper' and no extension method 'ToUpper' accepting a first argument of type 'int' could be found (are you missing a using directive or an assembly reference?)

Dynamics to the rescue


```
dynamic lang = "C#";
lang++;

dynamic theAnswer = 42;
theAnswer.ToUpper();
```

Dynamics to the rescue


```
dynamic lang = "C#";
lang++;

dynamic theAnswer = 42;
theAnswer.ToUpper();
```

Dynamics to the rescue


```
dynamic lang = "C#";
lang++;

dynamic theAnswer = 42;
theAnswer.ToUpper();
```

What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object calc = new Calculator();
int sum = calc.Add(10, 20);
```

What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object calc = new Calculator();
int sum = calc.Add(10, 20);
```

What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object calc = new Calculator();
int sum = calc.Add(10, 20);
```

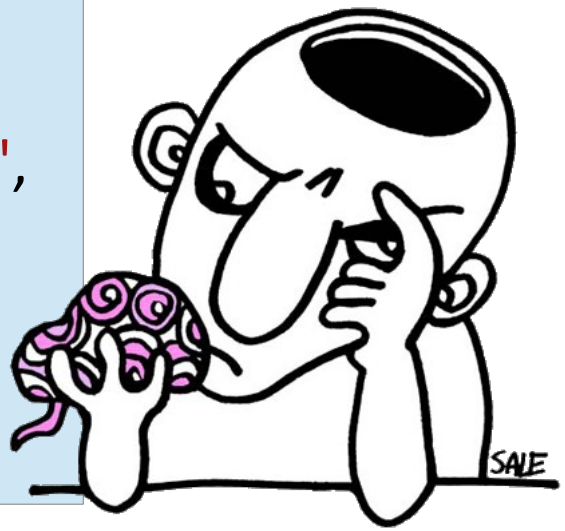
```
object reflectionCalc = new Calculator();
Type calcType = reflectionCalc.GetType();
object result = calcType.InvokeMember("Add",
      BindingFlags.InvokeMethod, null,
      Activator.CreateInstance(calcType),
       new object[] { 10, 20 });
int sum2 = Convert.ToInt32(result);
```

What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object reflectionCalc = new Calculator();
Type calcType = reflectionCalc.GetType();
object result = calcType.InvokeMember("Add",
      BindingFlags.InvokeMethod, null,
      Activator.CreateInstance(calcType),
       new object[] { 10, 20 });
int sum2 = Convert.ToInt32(result);
```

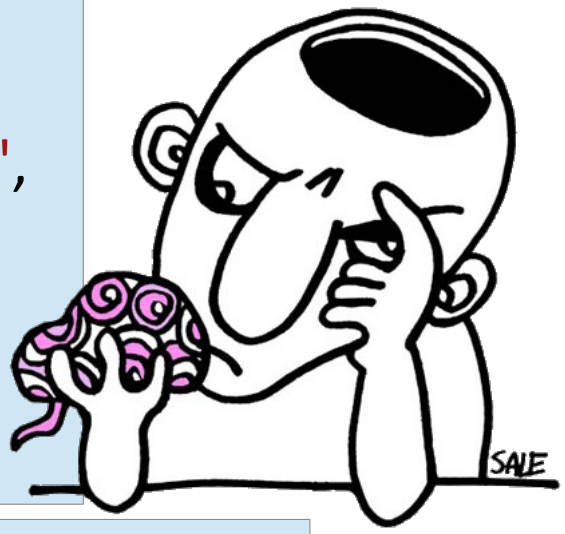


What about 'object' or reflection?


```
Calculator calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object calc = new Calculator();
int sum = calc.Add(10, 20);
```

```
object reflectionCalc = new Calculator();
Type calcType = reflectionCalc.GetType();
object result = calcType.InvokeMember("Add",
      BindingFlags.InvokeMethod, null,
      Activator.CreateInstance(calcType),
       new object[] { 10, 20 });
int sum2 = Convert.ToInt32(result);
```

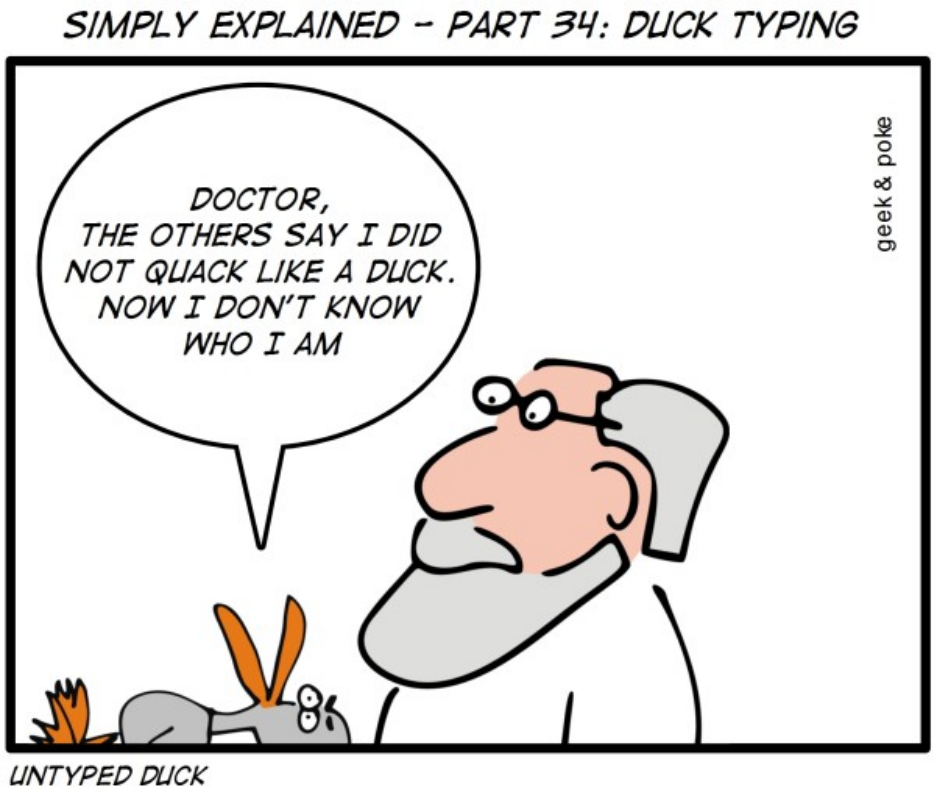



```
dynamic calc = new Calculator();
int sum = calc.Add(10, 20);
```

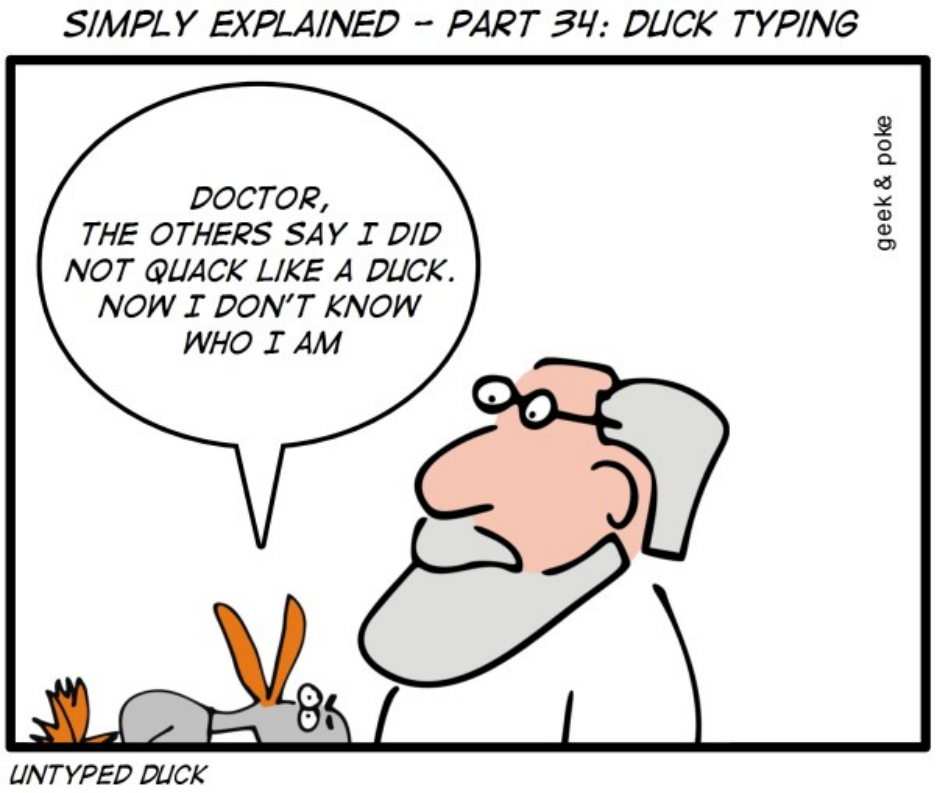
Duck-Typing

Duck-Typing

Duck-Typing


SIMPLY EXPLAINED – PART 34: DUCK TYPING
DOCTOR,
THE OTHERS SAY I DID
NOT QUACK LIKE A DUCK.
NOW I DON'T KNOW
WHO I AM
geek & poke
UNTYPED DUCK

Duck-Typing



- When I see a bird that walks like a duck and swims like a duck and quacks like a duck, I call that bird a duck

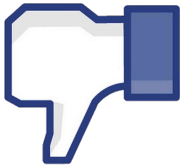
  James Whitcomb Riley

- Methods & Attributes > Class

Drawbacks

Drawbacks

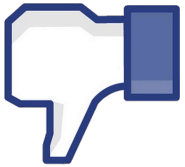
- No Compile time checks



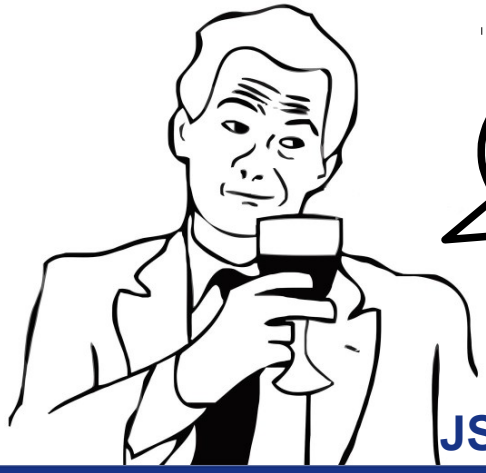
- Dynamics
  - → No member inference
  - → No IntelliSence

Drawbacks

- No Compile time checks

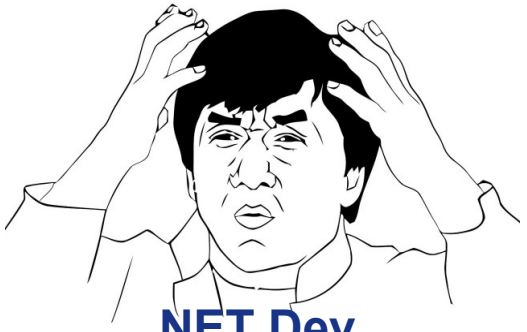


- Dynamics
  - → No member inference
  - → No IntelliSence



Welcome Bro

WAIT, WAT?



JS/Ruby/... Dev

.NET Dev

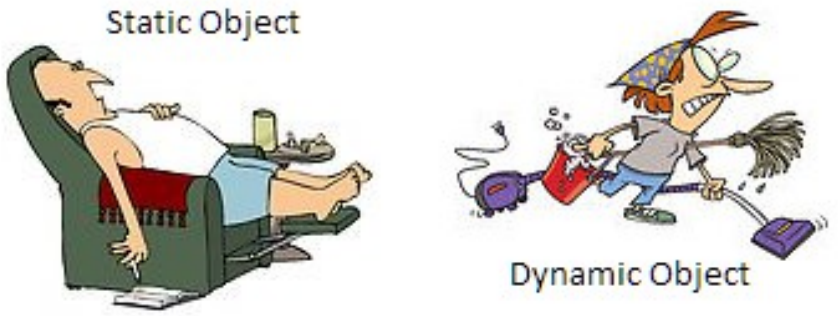
Dynamic languages – IronPython (or IronRuby)

```
#Python script.py
def add(a, b):
   return a + b
```

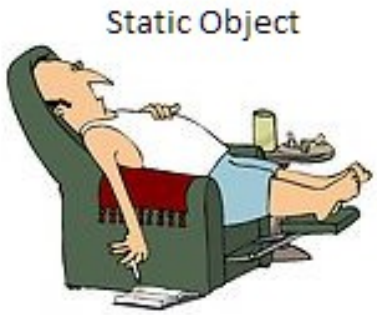
```
var pythonRuntime = Python.CreateRuntime();
dynamic pythonScript =
      pythonRuntime.UseFile("script.py");
var result = pythonScript.add(100, 200)));
```

Tools

DynamicObject

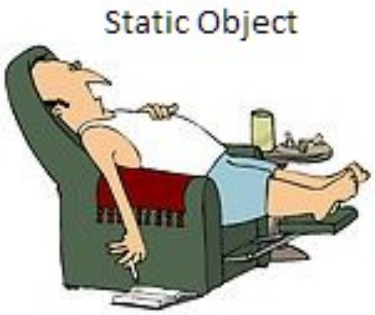


DynamicObject



ExpandoObject

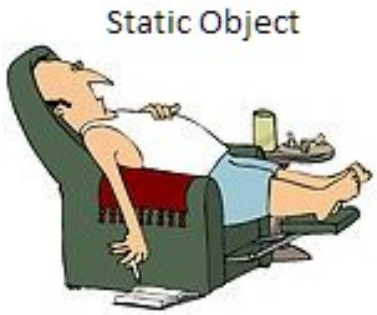
DynamicObject



ExpandoObject

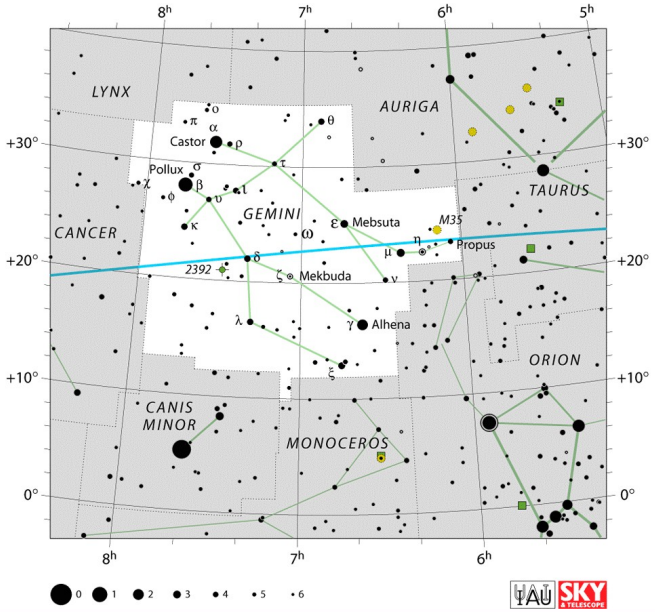
ElasticObject

DynamicObject



ExpandoObject

ElasticObject



Gemini

Frameworks

Massive

Massive

NANCY

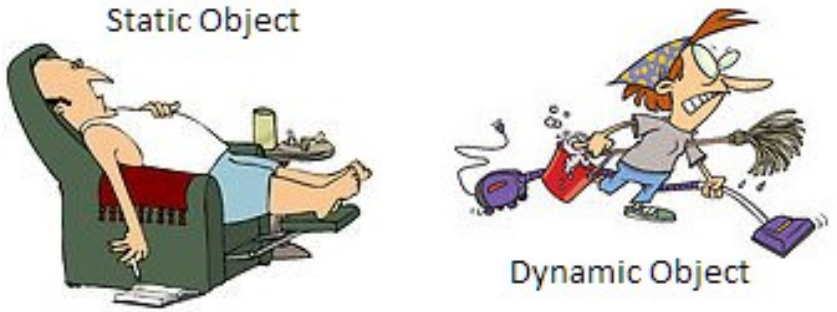
Massive

NANCY

Simple.Data

System.Dynamic.DynamicObject

- Exposes members at run time instead of at compile time



- Important methods
  - TrySetMember
  - TryGetMember
    - Is called when a member of a dynamic class is requested and no arguments are specified
  - TryInvokeMember
    - Is called when a member of a dynamic class is requested with arguments
- Combining those functions in a smart way is the key

System.Dynamic.ExpandoObject

- Represents an object whose members can be dynamically added and removed at run time
- Demo
  - Simple ExpandoObject
  - Expando structure vs Xml structure
  - ExpandoToXml
  - Linq-to-Object

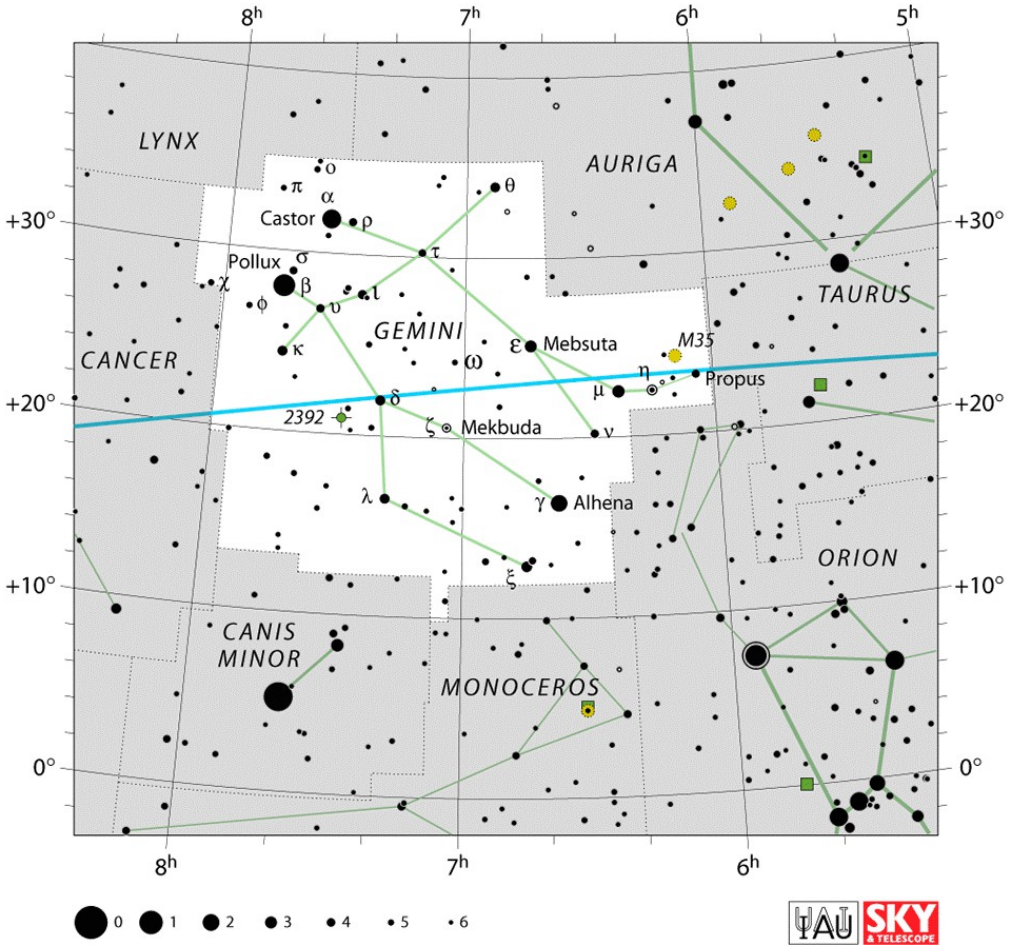
http://blogs.msdn.com/b/csharpfaq/archive/2009/10/01/dynamic-in-c-4-0-introducing-the-expandoobject.aspx

ElasticObject

- Multi level dynamic object implementation using .NET 4.0 dynamic features, for fluent access of data types like XML
- Demo
  - Expando vs Elastic
  - Elastic-to-Xml

https://github.com/amazedsaint/ElasticObject

- „Brings the capabilities of a dynamic type system to C#“
- Demo
  - Members on the fly
  - Methods on the fly
  - Object graph
  - Responds to
  - Introspection



- Lightweight WebFramework
- Demo
  - Parameters
  - Return object

NANCY

Massive

- Wrapper for DB tables that uses dynamics
- Create a class that wraps a table
- Query away
- Demo
  - Usage
  - Definition of TryGetMember

Simple.Data


I THINK WE SHOULD BUILD AN SQL DATABASE.
UH-OH
DOES HE UNDERSTAND WHAT HE SAID OR IS IT SOMETHING HE SAW IN A TRADE MAGAZINE AD?
WHAT COLOR DO YOU WANT THAT DATABASE?
I THINK MAUVE HAS THE MOST RAM.
E-mail: SCOTTADAMS@AOL.COM
© 1995 United Feature Syndicate, Inc.(NYC)

- A lightweight, dynamic data access component for .NET, written in C#
- Demo
  - TrimPathApi
  - Simple.Data

Conclusion

- Objects
  - Core Objects: DynamicObject, ExpandoObject
  - Variations: ElasticObject, Gemini
  - Usages: NancyFx, Massive, Simple.Data
- DTOs
- Architectural Flexibility
- API Design
- Think about using it!