BASTA! ®

Share Connect 2010
Die Konferenz für SharePoint Development & Management

SQLCON 2010

Oliver Sturm | www.oliversturm.com

Dynamic C# 4.0 - The Provider Side

Oliver Sturm

MVP Microsoft Most Valuable Professional

ineta

thinktecture

Oliver Sturm (@olivers)

- **Consultant and Trainer**
- **Associate Consultant at thinktecture**

- **.NET Application Architecture**
  - User Interfaces
  - Data Handling / Data Access Architectures
  - Programming Languages
  - DevExpress Component/Framework Products

- **Microsoft MVP for C#**
- **INETA Europe Speaker**

- **Services: http://www.oliversturm.com**
- **Blog: http://www.sturmnet.org/blog**

- **oliver@oliversturm.com**

Agenda

- Dynamic support in .NET 4 and C# 4 - there and back

- Basics of dynamic classes in C#

- Dynamic DB access - a slightly more complex example

- Providing dynamic types to IronPython

- Finally: should you really be dynamic? If so, when?

There and back

CLR / C#

Dynamic world

python™

```
dynamic thing = ...;
thing.GoDoIt();
```

→ DLR →
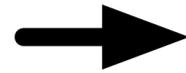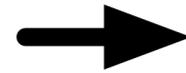
```
class Thing:
    def GoDoIt(self):
        ChangeWorld()
```

There and back

CLR / C#   **Dynamic world**

python™

```
dynamic thing = ...;
thing.GoDoIt();
```

```
class Thing:
    def GoDoIt(self):
        ChangeWorld()
```

DLR

```
class CSharpThing :
  DynamicObject {
  ...
}
```

```
thing = CSharpThing()
thing.DoItYourself()
```

There and back

CLR / C#   **Dynamic world**

python™

```
dynamic thing = ...;
thing.GoDoIt();
```

→ DLR →
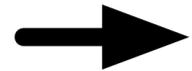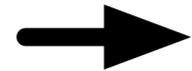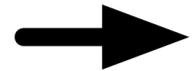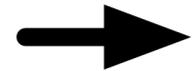
```
class Thing:
    def GoDoIt(self):
        ChangeWorld()
```

```
class CSharpThing :
  DynamicObject {
  ...
}
```

← DLR ←

```
thing = CSharpThing()
thing.DoItYourself()
```

↑

```
dynamic csthing =
  new CSharpThing();
csthing.DoItYourself();
```

Dynamic Classes in C#

○ IDynamicMetaObjectProvider

DynamicObject
DynamicObject
GetDynamicMemberNames
GetMetaObject
TryBinaryOperation
TryConvert
TryCreateInstance
TryDeleteIndex
TryDeleteMember
TryGetIndex
TryGetMember
TryInvoke
TryInvokeMember
TrySetIndex
TrySetMember
TryUnaryOperation

«type» **IDynamicMetaObjectProvider**
GetMetaObject

DynamicMetaObject
Expression
Value
...
BindBinaryOperation
BindConvert
BindCreateInstance
BindDeleteIndex
BindDeleteMember
BindGetIndex
BindGetMember
BindInvoke
BindInvokeMember
BindSetIndex
BindSetMember
BindUnaryOperation
Create
DynamicMetaObject
GetDynamicMemberNames

Demo

Basics of creating dynamic classes in C#

The dynamic database

Providing dynamic types to IronPython

Should you be dynamic?

- What makes a class (need to be) dynamic?

  - Look at the members - is there at least one (or a group of members) that isn't fully defined at compile time?

- Even if a class should be dynamic, that doesn't have to apply to all members!

- On that basis, let's have another look at DynamicField...

Demo

Taking the dynamic out of DynamicField

Summary

- Creating dynamic types in C# is perfectly possible

- Dynamic types can be used within C# itself and in dynamic (DLR) languages

- Doing it in a dynamic language directly is almost certainly easier

- C# may be a "central point" to provide dynamic types

Thank you

Please feel free to contact me about the content anytime.

oliver@oliversturm.com

Wrox Programmer to Programmer™

Functional Programming in C#
Classic Programming Techniques for Modern Projects

Oliver Sturm

wrox

Updates, source code, and Wrox technical support at www.wrox.com