Faculty Summit 2010
Introduction to Dynamism on .NET

Judith Bishop
Microsoft Research
<table>
<thead>
<tr>
<th>Year</th>
<th>Language 1</th>
<th>Language 2</th>
<th>Feature 1</th>
<th>Feature 2</th>
<th>Feature 3</th>
<th>Feature 4</th>
<th>Feature 5</th>
<th>Feature 6</th>
<th>Feature 7</th>
<th>Feature 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001</td>
<td>C# 1.0</td>
<td>F#</td>
<td>Spec#1.0</td>
<td>Spec#1.0</td>
<td>C# 2.0</td>
<td>C# 3.0</td>
<td>Code</td>
<td>Python</td>
<td>Code</td>
<td>Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>.5</td>
<td>.5</td>
<td>Spec#1.0</td>
<td>Spec#1.0</td>
<td>Contracts</td>
<td>3.0</td>
<td>Contracts</td>
<td>1.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.0.6</td>
<td>1.0.6</td>
<td></td>
<td></td>
<td></td>
<td>F# in VS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ruby on</td>
<td>Ruby on</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rails</td>
<td>Rails</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td></td>
<td></td>
<td>Java 1.5</td>
<td>Java 1.5</td>
<td>Java 6</td>
<td>Java 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LINQ</td>
<td>LINQ</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2008</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2009</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **C# 1.0**
- **F#**
- **Java 1.5**
- **C# 2.0**
- **Spec# 1.0.6**
- **Ruby on Rails**
- **Java 6 LINQ**
- **C# 3.0**
- **Code Contracts**
- **Python 3.0**
- **C#4.0 Code Contracts 1.4 F# in VS**

- **IE6**
- **Safari 1**
- **Firefox 2**
- **IE7**
- **Firefox 3**
- **Safari 4**
- **IE8**
- **Safari 5**

- **Windows XP**
- **.NET**
- **Mac OS X**
- **Rotor**
- **Mono 1.0**
- **Ubuntu Linux**
- **.NET 2**
- **Windows Vista**
- **Rotor 2.0**
- **Mac OS X Intel**
- **.NET 3.5**
- **Mac OSX Leopard**
- **DLR beta**
- **Windows 7**
- **.Net 4.0**
- **Mac OS XSnow.**
- **DLR 1.0**

- **Eclipse 1.0**
- **VS 2003**
- **Eclipse 3.0**
- **VS 2005**
- **VS 2008**
- **Eclipse 3.6**
- **VS2010**
Advantages
• Rapid feedback loop (REPL)
• Simultaneous top-down and bottom-up development
• Rapid refactoring and code changing
• Easy glue code

Uses
• Scripting applications
• Building web sites
• Test harnesses
• Server farm maintenance
• One-off utilities or data crunching
# The Development of C#

<table>
<thead>
<tr>
<th>C# 1.0 2001</th>
<th>C# 2.0 2005</th>
<th>C# 3.0 2007</th>
<th>C# 4.0 2009</th>
</tr>
</thead>
<tbody>
<tr>
<td>structs</td>
<td>generics</td>
<td>implicit typing</td>
<td>dynamic lookup</td>
</tr>
<tr>
<td>properties</td>
<td>anonymous</td>
<td>anonymous types</td>
<td>named and optional</td>
</tr>
<tr>
<td>foreach loops</td>
<td>methods</td>
<td>object and array</td>
<td>arguments</td>
</tr>
<tr>
<td>autoboxing</td>
<td>iterators</td>
<td>initializes</td>
<td>COM interop</td>
</tr>
<tr>
<td>delegates and</td>
<td>partial types</td>
<td>extension</td>
<td></td>
</tr>
<tr>
<td>events</td>
<td>nullable types</td>
<td>methods,</td>
<td></td>
</tr>
<tr>
<td>indexers</td>
<td>generic</td>
<td>lambda</td>
<td></td>
</tr>
<tr>
<td>operator</td>
<td>delegates</td>
<td>expressions</td>
<td></td>
</tr>
<tr>
<td>overloading</td>
<td></td>
<td>query expressions</td>
<td></td>
</tr>
<tr>
<td>enumerated types</td>
<td></td>
<td>(LINQ)</td>
<td></td>
</tr>
<tr>
<td>with IO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>in, out and ref</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>formatted output</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### API
- Serializable
- Reflection

- std generic delegates
The dynamic language runtime (DLR) is a runtime environment that adds a set of services for dynamic languages – and dynamic features of statically typed languages – to the common language runtime (CLR).
Dynamic Lookups
- Calls, accesses and invocations bypass static type checking and get resolved at runtime

Named, default and optional parameters

COM interop

Variance
- Extends type checking in generic types
- E.g. `IEnumerable<string>` is also `IEnumerable<object>`
• If d is a standard .NET object, and the operation will be dispatched using reflection on its type and a C# “runtime binder”

```csharp
dynamic d1 = GetDynamicObject();
dynamic d2 = new Bar();
string s;
d1.M(s, d2, 3, null);
```

• Compiled as: Perform an instance method call of M with second arg. dynamic
d• DLR has an fast caching mechanism to keep the reflection efficient
d• d’s class can implement the interface IDynamicObject and completely change the meaning of dynamic operations
d• The DLR is invisible unless you implement IDynamicObject
d• Used with IronPython and IronRuby (and C# and F#)
Welcome, Willkommen, Bienvenue

Welcome to the IronPython release and source repository page on CodePlex. Here you'll find downloads for all major releases and the latest source code for IronPython. If you're looking for information on IronPython such as documentation, samples, support, etc. we highly recommend visiting the official IronPython website, which not only contains documentation created by the IronPython Team, but also links to community generated content.

About

IronPython is an implementation of the Python programming language running under .NET and Silverlight. It supports an interactive console with fully dynamic compilation. It's well integrated with the rest of the .NET Framework and makes all .NET libraries easily available to Python programmers, while maintaining compatibility with the Python language.

IronPython is an open source project freely available under the Microsoft Public License. You can grab the latest sources by clicking on the Source Code tab at the top of this page or simply by clicking here. Using Team Explorer from inside Visual Studio will give a richer view of source history.

Thanks to all the users over the years who provided feedback that helped shape IronPython!

Download the latest stable release, run IronPython in your web browser, or grab older releases now!
IronPython is an open-source implementation of the Python programming language which is tightly integrated with the .NET Framework. IronPython can use the .NET Framework and Python libraries, and other .NET languages can use Python code just as easily.
IList<string> list1 = new List<string>();
IList<object> list2 = list1;

Disallowed so that e.g. an int cannot be put into list2, and then extracted from list1.

But when objects cannot be inserted, then assignment should be allowed

IEnumerable<object> list2 = list1;

Variance allows assignment to work when it is safe.
class Animal { }
class Cat : Animal { }
class Program {
    delegate T Func1<out T> ();
    delegate void Action1<in T>(T a);
    static void Main(string[] args) {
        // Covariance
        Func1<Cat> cat = () => new Cat();
        Func1<Animal> animal = cat;
        // Contravariance
        Action1<Animal> act1 = (ani) => {
            Console.WriteLine(ani);
            Action1<Cat> cat1 = act1;
        }
    }
}

• Support for method groups with matching method signatures with delegate types.
• Covariance
  • Preserves assignment compatibility
• Contravariance
  • Reverse it
• COM is not going away!
• Everything is passed by ref

```csharp
static void Main() {
    Word.Application app =
        new Word.Application();
    object m = Type.Missing;
    app.Documents.Add(ref m, ref m,
        ref m);
}

static void Main() {
    Word.Application app =
        new Word.Application();

    // GOOD
    app.Documents.Add(Type.Missing,
        Type.Missing, Type.Missing);

    // BEST
    app.Documents.Add();
}
```
JavaScript and Browsers

- **Dynamic**
  - dynamic typing
  - object based
  - run-time evaluation

- **Functional**
  - first-class functions
  - inner functions and closures

- **Prototype-based**
  - prototypes instead of classes for inheritance
  - functions as object constructors
  - functions as methods

- JavaScript is the only cross-browser language
- Non-language factors matter too

- Yermolovich, Wimmer and Franz 2009