

THE COBRA PROGRAMMING LANGUAGE

cobra-language.com
Lang.NET 2008

INTRO

- Cobra is a new language
- Object-oriented, imperative
- Embraces unit tests, contracts and more
- General purpose
- Runs on .NET & Mono
- Windows, Mac, Linux, Solaris, etc.

MOTIVATION

- Productivity boosters are scattered across languages
 - Clean syntax (Python, Ruby)
 - Run-time performance (C#, C++)
 - Static and dynamic typing (Objective-C, VB)
 - Contracts (Eiffel, Spec#)
 - Nil tracking (Spec#, iihtdioa.C#)
- Not mutually exclusive!

I WANT IT ALL

- No more jumping around
 - Clean syntax (Cobra, Python, Ruby)
 - Run-time performance (Cobra, C#, C++)
 - Static and dynamic typing (Cobra, Objective-C, VB)
 - Contracts (Cobra, Eiffel, Spec#)
 - Nil tracking (Cobra, Spec#, iihtdioa.C#)
- Goal is maximum productivity

NO NIL UNLESS I SAY SO

- NullReferenceExceptions happen *one at a time at run-time*
- Method sigs don't indicate if they return or accept it
- `def nodeFor(name as String) as Node?`
- `def nodeFor(name as String?) as Node?`
- Compile-time detection happens *many times at compile-time*

SQUEAKY CLEAN SYNTAX

- Python-like
- Light on symbols, indented blocks, keywords
- list literals, dict literals, (soon) set literals
- in / not in, is vs. ==
- But even cleaner!
 - Straight forward properties
 - Other tweaks

DYNAMIC OR STATIC? BOTH!

- Programmers should choose, not language designers
- Objective-C has been doing it for ~20 years
- `def add(a as int, b as int) as int`
- `def add(a, b) as dynamic`
- There are pros and cons to both
- Don't have to switch languages to switch approaches

DYNAMIC IS CLEARLY BEST!

- ```
def add(a, b) as dynamic
 return a + b
```
- Flexible
- Fast coding and prototyping
- Less brittle w.r.t. changes
- More reusable

# STATIC IS CLEARLY BEST!

- `def nodeFor(name as String) as INode?`
- Compile-time detection of errors
- Multiple errors reported at once
- Fast at run-time
- Slim too (no boxing)
- Easy Intellisense

# PERFORMANCE

- Performance can be very important
- ... financial analysis, video games, compilers, AI, ...
- Performance can *become* important
  - Yahoo Mail: Python, then C++
  - AI company: Ruby prototype, then C++
- Cobra compiles and leans towards static
- “`i = 5`” infers “`i`” as an “`int`”

# SCRIPTING CONVENIENCE

- Compile and run in one command:  
    > cobra foo.cobra
- Clean syntax is a hallmark of *some* scripting languages
- `#!` line on Unix-like systems
- Dynamic binding is a hallmark of scripting languages

# CONTRACTS

- ```
def nodeFor(name as String) as INode?  
  require name.length  
  ensure  
    result.name.toLowerCase == name.toLowerCase  
  ...
```
- Supports invariant, old, result and implies
- Inheritance works
- Eiffel-style: the “real thing”
- Future? Integrate with Spec# backend

UNIT TESTS

- `def capped(s as String) as String is shared test`
 - `assert Utils.capped('aoeu') == 'Aoeu'`
 - `assert Utils.capped('') == ''`
 - `expect NullPointerException`
 - `Utils.capped(nil) # ahem``body`
 - `...`
- Same motivations as doc strings:
 - localized, encourage use, get people on same page

ACCURATE MATH IN 2008

- 0.1 added ten times is what?
In most languages: not 1.0!
- Python:

```
>>> .1+.1+.1+.1+.1+.1+.1+.1+.1+.1
0.999999999999999999999989
>>> assert 1.0 == .1+.1+.1+.1+.1+.1+.1+.1+.1+.1
AssertionError
```
- Cobra supports both decimal and float (64-bit)
- Defaults to decimal because it's 2008

THE COMPILER

- Self-implemented a.k.a “self-hosted”
- Usual phases:
tokenize, parse, AST nodes, analysis, code gen
- Something different: chose C# as backend over IL
 - Growing number of “super-VM” features in C#
 - Faster implementation
 - Piggy back on error checking and cmd line options

VEND TO C# AND VB

- You can vend class libraries to C# and VB, both technically and practically.
- Super-C# features like non-nil degrade gracefully
- Technically: .NET/Mono DLLs and CLI-style classes
- Practically
 - Cobra favors .NETisms like generic lists
 - Can embed Cobra run-time (no Cobra.Lang.dll)

WEAKNESSES

- Maturity - still gaps and some bugs
- More nifty features not implemented than I would prefer (upcoming slide)
- No IDE plug-ins
- No interactive prompt

FEBRUARY 2008!

- An exciting month for Cobra!
- Leaving “stealth mode”
- Open sourcing the compiler
- Discussion forums
- Wiki
- Issue tracker

COMMERCIALISM

- In 2007, I worked full time on Cobra.
Paid rent with savings (and a poker tournament).
- In 2008, return to contracting.
Work on Cobra part time. :-)
- Ideas:
 - Visual Cobra / VS PlugIn
 - Book, Web site ads
 - Microsoft | Novell sponsors Cobra :-)

FUTURE FEATURES

- Context: Be the best, most productive, high-level, general-purpose OO language.
- Full LINQ and friends (lambdas, etc.)
- Language level reg-ex
- Built-in Set
- mix-ins / traits / ...
- DLR integration

MORE FUTURE FEATURES

- Units of measurement (feet, meters, ...)
- Compile-time analysis of contracts

```
def foo(thing)
  require
    thing responds to (get name as String)
```
- Multiple backends
JVM, Objective-C, LLVM, Parrot, ...

THE FAR FUTURE

- Parallel programming
- Futures / lazy arguments
- Macros
- Would be nice to leverage .NET advances as with generics, LINQ, etc.

WEB SITE

- cobra-language.com
- cobra-language.com/docs/why
- cobra-language.com/docs/python
- Sample programs, How To, Documentation
- cobralang.blogspot.com
- Chuck.Esterbrook@gmail.com