VisualBasic and the Java™ Platform

Herbert Czymontek
John Kline
Tor Norbye

Senior Staff Engineers
Sun Microsystems, Inc.

TS-3576

Code Name

Project Semplice

New Code Name?

einfacheproprammiersprache

Warning—Subject to change

Agenda

Overview

Language Design Principles

Language Features

Performance

Tooling

Demo

Project Semplice BASIC—What Is It?

- BASIC
- Inspired by VisualBasic
- Compiles to Java™ VM .class files
- Interoperates with Java technology and Java APIs
- Compatible with JSR 223 scripting engine framework
- Performance and productivity included!

Agenda

Overview

Language Design

Language Features

Performance

Tooling

Demo

Language Design Philosophy

- Seek developer feedback early and often
- Ease of use over features and standards
  - When in conflict
- Attract new programmers
  - Those not comfortable with Java language and OOP
- Attract VB programmers
  - looking for a “simple” BASIC on the Java VM

Language Design Philosophy (Cont.)

- Use of the new features is optional
  - inheritance, exception handling, threading
- The language is not enough
  - Need first class tools support
  - Simplify programming paradigm

Ease of Use Goes to 11

Project Semplice BASIC: An easy to learn language and programming paradigm

Sneak Peek

- Import a VB project into NetBeans™ IDE
- Build it
- Run it

Agenda

Overview

Language Design

Language Features

Performance

Tooling

Demo

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit Integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

32-bit Integer/64-bit Long

	VB6	Our BASIC	VB.Net
Boolean	8-bit	8-bit	8-bit
Byte	8-bit	8-bit	8-bit
Char	N/A	16-bit	16-bit
Short	N/A	16-bit	16-bit
Integer	16-bit	32-bit	32-bit
Long	32-bit	64-bit	64-bit
Float	32-bit	32-bit	32-bit
Double	64-bit	64-bit	64-bit

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and Imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Unicode Strings

- Project Semplice BASIC strings are just like Java language strings
- Characters in strings are Unicode characters

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Simple OOP

- Support for Java technology style single inheritance and interfaces
- Some new keywords:
  - `Class`
  - `Inherit`
  - `Module`

Code Sample—Simple OOP

```
Class Foo
    Sub Hello
        java.lang.System.out.println “Hello Foo!”
    End Sub
End Class

Class Bar Inherit Foo
    Sub Hello
        java.lang.System.out.println “Hello Bar!”
    End Sub
End Class

Module FooBar
    Sub Main(p As Foo)
        p.Hello
    End Sub
End Module
```

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and Imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Namespaces and Imports

- Namespaces are the BASIC equivalent of packages in the Java programming language

```
Namespace com.sun.semplice
End Namespace
```

- Imports in BASIC work pretty much like imports in the Java language

```
Imports java.lang
Imports java.util.Vector
```

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Structured Exception Handling

- Very similar to exception handling in the Java language

```
Try
...
Catch ex As Exception
...
Finally
...
End Try
```

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer / 64-bit long
- Unicode strings
- Simple OOP
- Namespaces and imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Threading and Synchronization

- Project Semplice BASIC uses Java language threads
- New keyword for synchronization support:

```
SyncLock obj
      ...
End SyncLock
```

VisualBasic vs. Project Semplice BASIC

What are the differences?

- 32-bit integer/64-bit long
- Unicode strings
- Simple OOP
- Namespaces and imports
- Structured exception handling
- Threading and synchronization
- Connecting with the Java technology world

Connecting With the Java Technology World

- Identifier character sets
  - Problem: In VB identifiers are limited to the ASCII character set (without '_')
  - Solution: Not really a problem, we just change the rules and allow all characters legal for Java language identifiers
- Case sensitivity:
  - Problem: In Basic *Add* and *add* are the same identifier
  - Solution: Again, not really a problem. Existing VB code is properly capitalized (VB IDE does that) and we change the rule: From now on identifiers (with the exception of keywords) are case sensitive

Connecting With the Java Technology World

- Method overloading
  - Problem: In Java technology it is allowed to have add() and add(int)
  - Solution: It is the developers responsibility to properly cast the actual method parameters so that there is exactly one matching method
- Type compatibility
  - Problem: BASIC String == Java language String?
  - Solution: Yes—There will be a mapping of all intrinsic BASIC types to Java technology types

Connecting With the Java Technology World

- Array compatibility
  - Problem: In BASIC arrays don’t have to be zero-based
  - Solution: There will be a conversion to re-base an array if it needs to be passed to a Java language method

Inside Project Semplice BASIC

- Type conversions
- Early vs. late binding
- Running programs with errors

Type Conversions

- Type conversion rule (Simple version):
  - Find the widest type of the involved operands, convert all operands to that type, apply the operation, find most narrow type that can represent the result and convert the result to that type.
- Example:

```
CelsiusTextBox.Text = (FahrenheitTextBox.Text – 32) / 1.8

      <String>                         <String>      <int> <float>
```

Inside Project Semplice BASIC

- Type conversions
- Early vs. late binding
- Running programs with errors

Early vs. Late Binding

- TS-3886 Dynamically Typed Languages on the Java™ Platform by Gilad Bracha
- Problem in a Nutshell:

```
Dim obj As Object
...
obj.foo “Hello”
```

- Two solutions:
  - Resolution at runtime via reflection
  - Use of new bytecode invokedynamic

Inside Project Semplice BASIC

- Type conversions
- Early vs. late binding
- Running programs with errors

Running Programs With Errors

- Builds class file regardless of whether any compile time errors were reported
- Reports a runtime error if execution reaches the error point

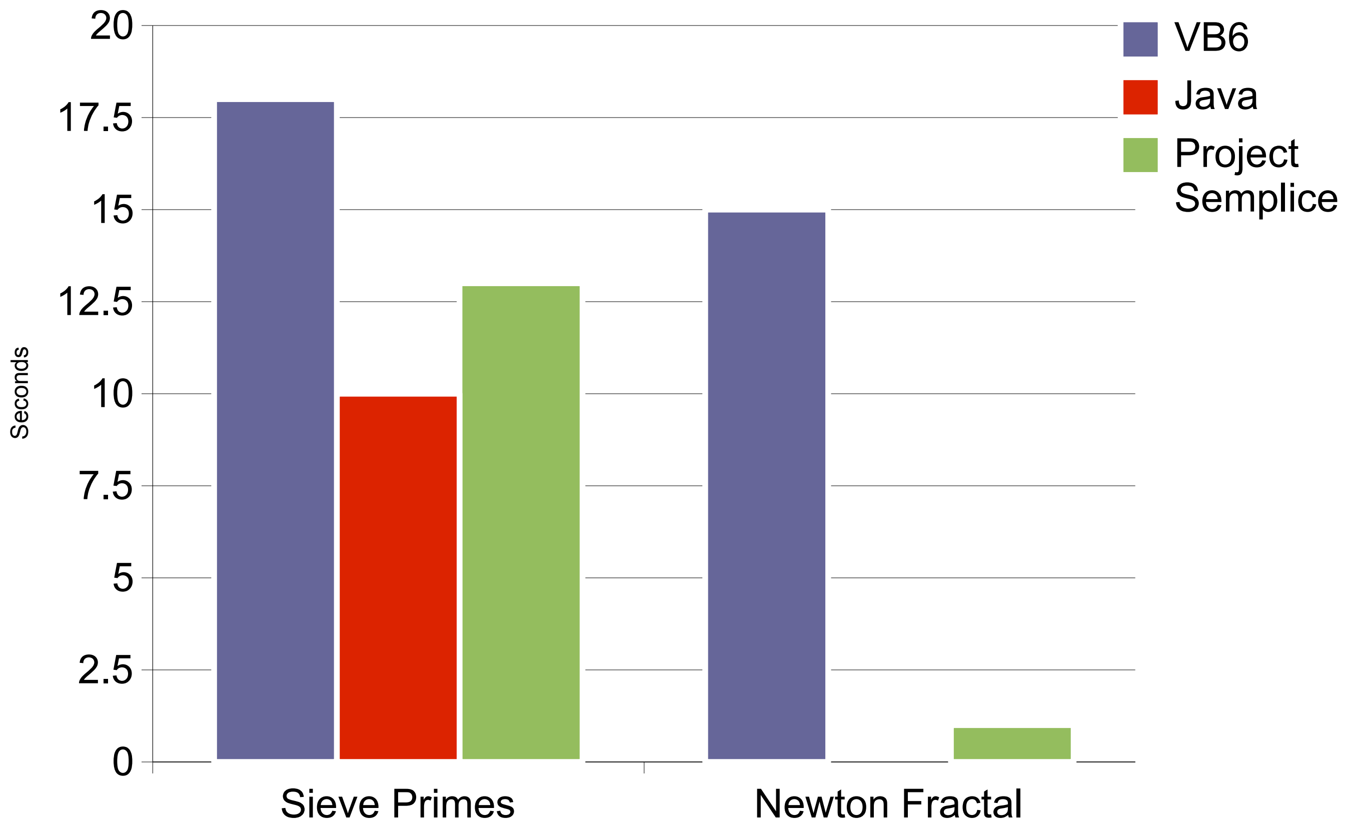
Agenda

- Overview
- Language Design
- Language Features
- Performance
- Tooling
- Demo

Performance

- Optional typing
  - If specified, same bytecode as Java
- Performance vs. VB 6
  - Demos will show significant benefit of Java HotSpot™ technology

Benchmarks



Source: Measured on Herbert's computer

Agenda

- Overview
- Language Design
- Language Features
- Performance
- Tooling
- Demo

Tools

- Full support in all of our tools
  - NetBeans IDE
  - Sun Java Studio Creator IDE
  - Sun Java Studio Enterprise software
- Full support = Equivalent to Java language support
  - Code completion, navigation, refactoring, quick fixes, …
  - Automatic case correction

RAD Tools

- Main goal is RAD support with components
  - Build web apps
  - Build GUI apps
  - …and use BASIC as the “glue”
- Shown in demos

Scripting Tools

- Our BASIC is javax.script.ScriptEngine enabled (JSR 223—see TS–1382)
- Your application can be scripted
  - Embed the BASIC Scripting Engine
  - Your users can write scripts
  - These scripts can drive your (Java technology) application
- BASIC IDE support will make this integration trivial

DEMO

Project Semplice in Action

Summary

- Project Semplice BASIC is:
    - Easy to learn and productive to use
    - Will be supported across Sun tools
    - Performant
    - Allows VB programmers an easy transition to the Java VM and APIs

Project Semplice BASIC

Leveraging the Java Platform
Without the Learning Curve

Project Semplice BASIC Will Improve Your Life

It's a floor wax and a dessert topping!

Q&A

Herbert Czymontek
John Kline
Tor Norbye

VisualBasic and the Java™ Platform

Herbert Czymontek
John Kline
Tor Norbye

Senior Staff Engineers
Sun Microsystems, Inc.

TS-3576