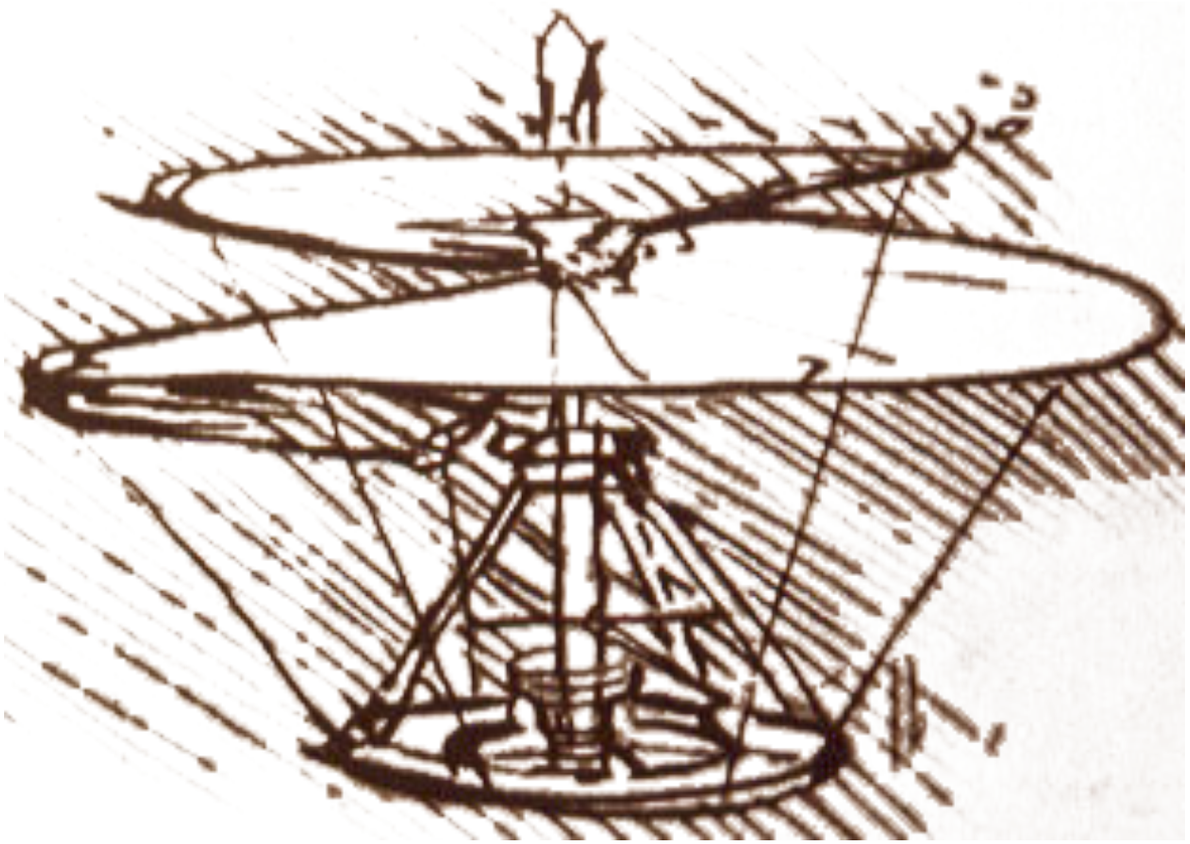
ThoughtWorks®

Java as a Platform



Ola Bini

computational metalinguist

ola.bini@gmail.com

http://olabini.com/blog

Your host

From Sweden to Chicago through ThoughtWorks

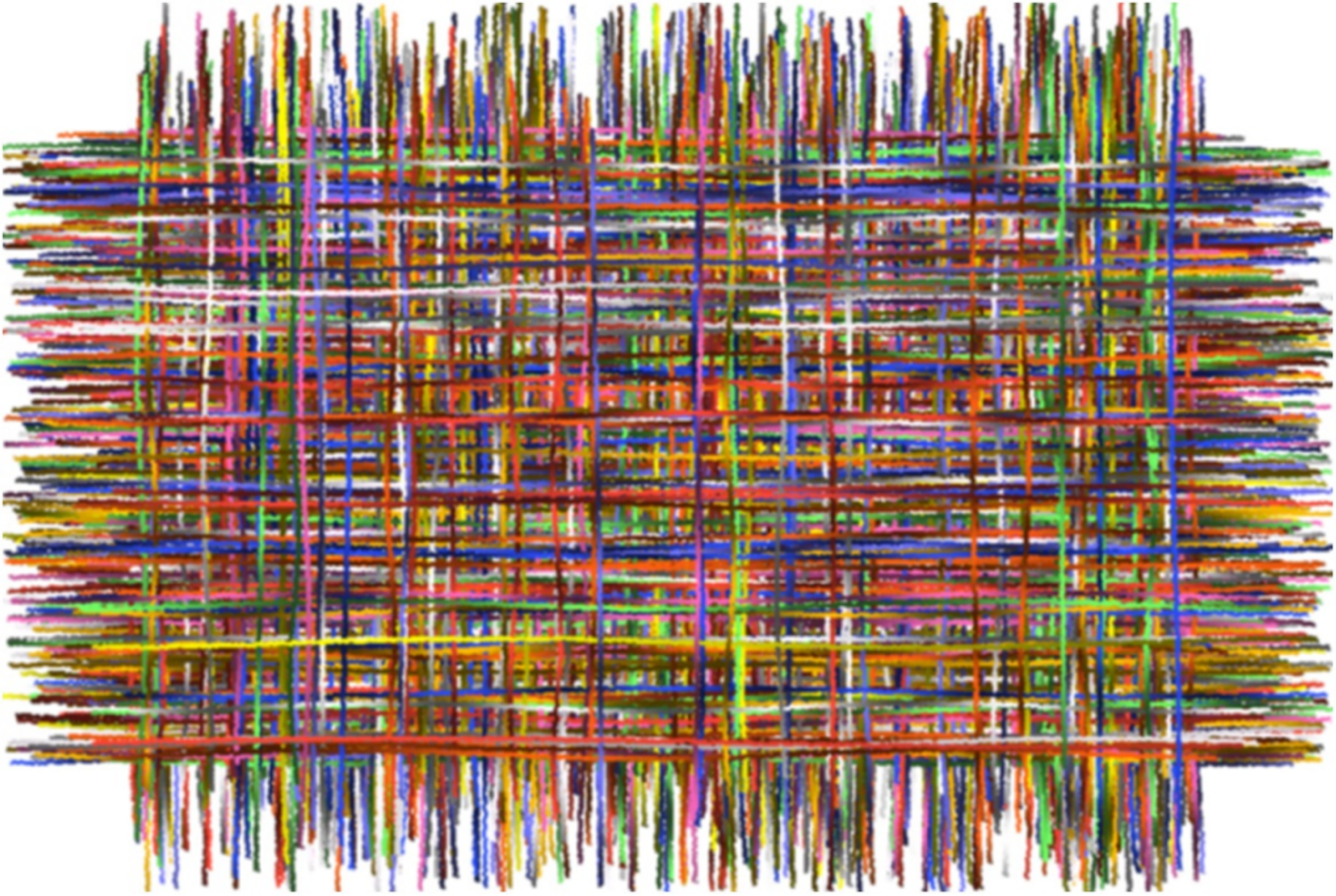
Language geek at ThoughtWorks

JRuby core developer, Ioke and Seph creator

Member of the JSR 292 EG

Platform

What you get



Code loading

Platform

independence

A CRYPTO NERD'S IMAGINATION:

HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.

NO GOOD! IT'S 4096-BIT RSA!

BLAST! OUR EVIL PLAN IS FOILED!

WHAT WOULD ACTUALLY HAPPEN:

HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.

GOT IT.

JIT

torsdag den 12 maj 2011

Languages

Aardappel
ABCL
AJLogo
Anvil
BDC Scheme
BeanShell
Bex Script
Bigloo
Bistro
CAL
Ceylon
CKI Prolog
Clojure
COCOA
CONVERT
Correlate
Demeter/Java
dSelf
Fan
FScript
Funnel
Gosu
Groovy
Hecl
Hojo
HotScheme
HotTEA
Ioke
iScript
Jacl
Jaja
Janino
Jatha
javalog
JBasic
Jickle
JLog
JMatch
Join Java
JoyJ
JRuby
JScheme
JudoScript
Jython
Kawa
Lili
Lisp
LL
Mapyrus
MetaJ
Mini
NetLogo
Nice
Obol
PERCobol
PLAN
Pnuts
PS3i
Quercus
Resin
Rhino
rLogo
Sather
Scala
SISC
Sixx
Skij
Sleep
SmallWorld
StarLogo
Talks2
TermWare
Thorn
tuProlog
Turtle Tracks
uts
v-language
W4F
webLISP
WLShell
XProlog
Yassl
Ync/Javascript
Yoix

Paradigms

Object Oriented

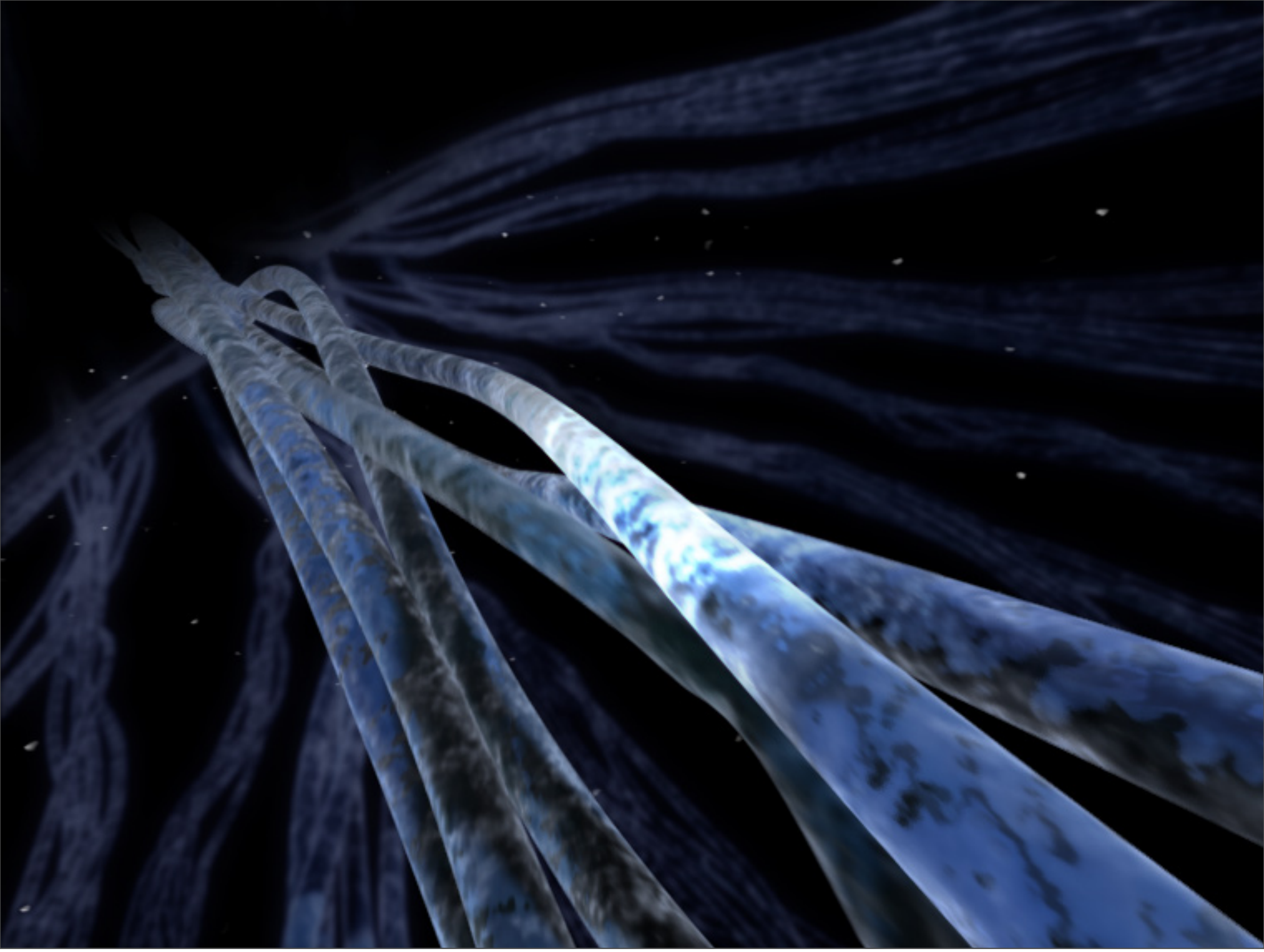
Statically typed

Dynamically typed

Concatenative

And many more...

What's missing?



Light code loading

Variable

static typing

Value objects

JVM Safepoint

JVM Library Help

Symbolic freedom

Interface Injection

Language Specific Invocation

The DaVinci Machine

JSR292

Dynamic

dispatch today

Some machinery

Class Values

Analog to ThreadLocal

Allows association of classes to values

Values are computed lazily

Installation of new values happen atomically

It works on any type, including int.class or void.class

You can remove Class Values for a specific type too

```
import java.lang.ClassValue;

public class ClassValueTest {
    private final static ClassValue<MethodTable> methods =
        new ClassValue<MethodTable>() {
            public MethodTable computeValue(Class<?> type) {
                return MethodTable.newTableFor(type);
            }
        };

    public static void main(String[] args) {
        MethodTable intMethods = methods.get(int.class);
        MethodTable stringMethods = methods.get(String.class);
        // Do something extremely cool here
    }
}
```

Method Handles

Combinators

Collect arguments

Spread arguments

Binding to a specific object

Convert a MethodHandle to an instance of a SAM

Convert arguments

Filter arguments and return values

Fold arguments

Catch exceptions

Permute arguments

Guard with test

static MethodHandle guardWithTest(MethodHandle test,
MethodHandle target,
MethodHandle fallback)

Easy way to implement Monomorphic Inline Cache

Most likely thing to be returned from InvokeDynamic

SwitchPoint

MethodHandle guardWithTest(MH target, MH fallback)

static void invalidateAll(SwitchPoint[] switchPoints)

Makes it possible to publish a one time state transition

It begins in a valid state

Can at any point become invalid

Invalidation is global, immediate and permanent

InvokeDynamic

Call site

Bootstrap Method

```
public static CallSite bootstrapOne(
            MethodHandles.Lookup callerClassLookup,
            String methodName,
            MethodType typeDescriptor,
            String staticArg1,
            String staticArg2) {
    return new ConstantCallSite(someMethodHandle);
}
```

InvokeDynamic

An Example

SephCallSite.java

```
public class SephCallSite extends MutableCallSite {
    private enum Morphicity {
        NILADIC, MONOMORPHIC, POLYMORPHIC, MEGAMORPHIC
    }

    int numberOfGuards = 0;
    Morphicity morphicity = Morphicity.NILADIC;

    public SephCallSite(MethodType type) {
        super(type);
    }
}
```

AbstractionCompiler.java

```
String bootstrapName = "basicSephBootstrap";
boolean fullPumping = false;

if(first) {
    if(current == last) {
        bootstrapName = "noReceiverTailCallSephBootstrap";
        fullPumping = true;
    } else {
        bootstrapName = "noReceiverSephBootstrap";
        fullPumping = false;
    }
} else if(current == last) {
    bootstrapName = "tailCallSephBootstrap";
    fullPumping = true;
}

ma.dynamicCall(encode(current.name()), sigFor(arity),
            bootstrapNamed(bootstrapName + possibleIntrinsic));
```

Bootstrap.java

```
private static CallSite bootstrap(MethodHandles.Lookup lookup, String name,
                                  MethodType type, String bootstrapType) {
    SephCallSite site = new SephCallSite(type);
    MethodType fallbackType = type.insertParameterTypes(0,
                                  SephCallSite.class, String.class);
    MethodHandle fallback = MethodHandles.insertArguments(
                        findStatic(Bootstrap.class, bootstrapType, fallbackType),
                        0, site, decode(name));
    site.setTarget(fallback);
    return site;
}

public static CallSite basicSephBootstrap(MethodHandles.Lookup lookup,
                                          String name, MethodType type) {
    return bootstrap(lookup, name, type, "fallback");
}
```

Bootstrap.java (continued)

```
public static SephObject fallback(SephCallSite site, String name, SephObject receiver,
                                  SThread thread, LexicalScope scope, IPersistentList args) {
    SephObject value = receiver.get(name);
    if(null == value) {
        throw new RuntimeException(" *** couldn't find: " + name + " on " + receiver);
    }
    if(value.isActivatable()) {
        site.installActivatableEntry(receiver, null, value, -1);
        return value.activateWith(receiver, thread, scope, args);
    } else {
        site.installConstantEntry(receiver, null, value, -1);
    }
    return value;
}
```

Bootstrap.java (continued)

```
public static SephObject initialSetup_intrinsic_if(SephCallSite site, MethodHandle slow,
                                                   SephObject receiver, SThread thread,
                                                   LexicalScope scope, MethodHandle test,
                                                   MethodHandle then, MethodHandle _else)
    throws Throwable {
    MethodHandle guarded = thread.runtime.INTRINSIC_IF_SP.guardWithTest(
                                  INTRINSIC_IF_MH, replaceCompletely3(slow, site));
    site.setTarget(guarded);
    return (SephObject)guarded.invokeExact(receiver, thread, scope, test, then, _else);
}
```

Runtime.java

```
public final SwitchPoint INTRINSIC_IF_SP = new SwitchPoint();
public static void empty() {}
public static void invalidate(SwitchPoint sp) {
    SwitchPoint.invalidateAll(new SwitchPoint[] {sp});
}
public final MethodHandle INVALIDATE_IF = INTRINSIC_IF_SP.guardWithTest(
                INVALIDATE_MH.bindTo(INTRINSIC_IF_SP), EMPTY_MH);
public void checkIntrinsicAssignment(String name) {
    name = name.intern();
    try {
        if(name == "true") {
            INVALIDATE_TRUE.invokeExact();
        } else if(name == "if") {
            INVALIDATE_IF.invokeExact();
        }
    } catch(Throwable e) {}
}
```

JSR292.next?

Questions?

Ola Bini

ThoughtWorks®

http://olabini.com
obini@thoughtworks.com
@olabini