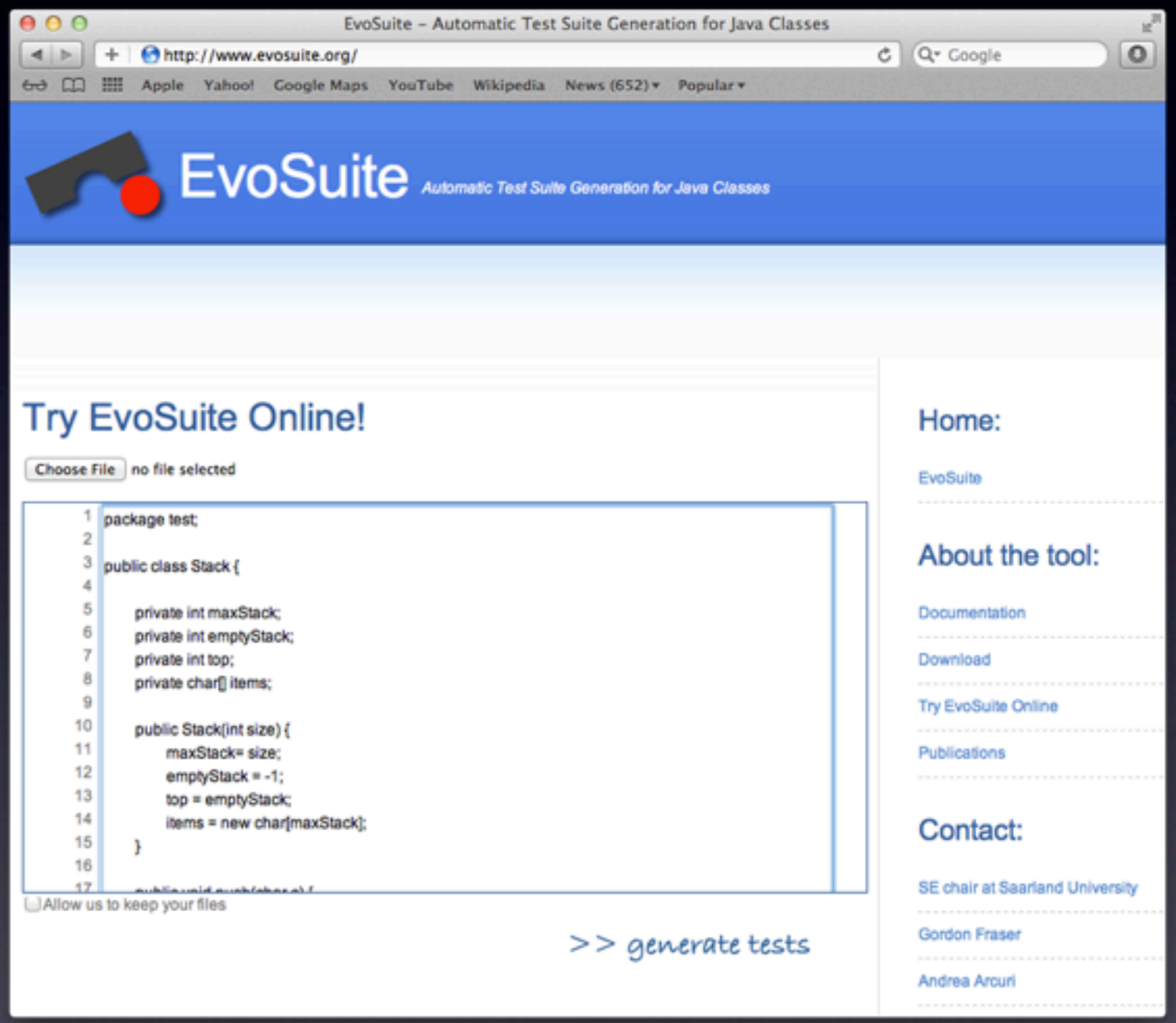
Bytecode Testability Transformation

Yanchuan Li and Gordon Fraser

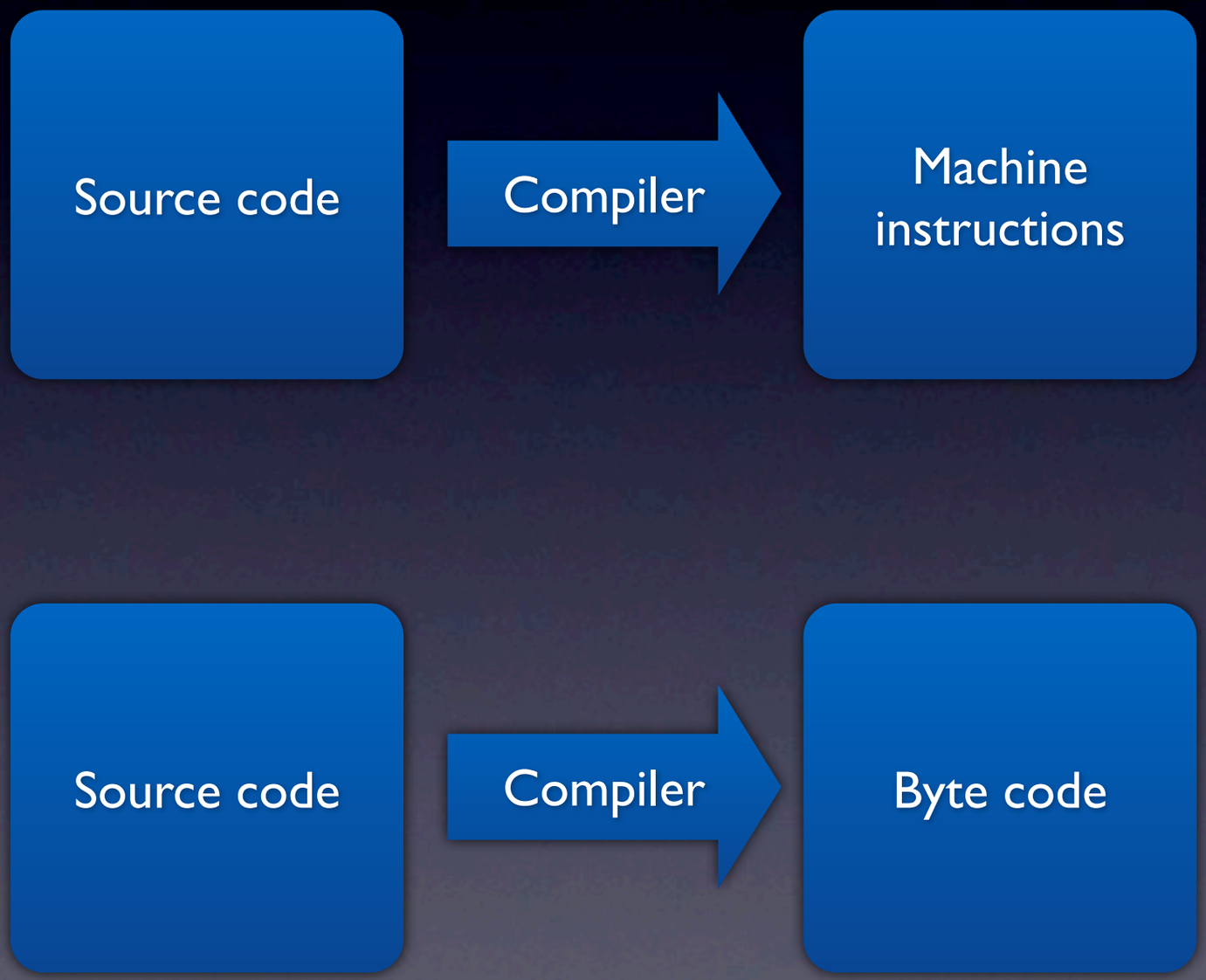
```
boolean flag = x < y;

if(flag) {
   // target branch
}
```

```
if(x < y) {
   // target branch
}
```


EvoSuite – Automatic Test Suite Generation for Java Classes
http://www.evosuite.org/
Google
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
News (652)
Popular
EvoSuite
Automatic Test Suite Generation for Java Classes
Try EvoSuite Online!
Choose File
no file selected
1 package test;
2
3 public class Stack {
4
5 private int maxStack;
6 private int emptyStack;
7 private int top;
8 private char[] items;
9
10 public Stack(int size) {
11 maxStack= size;
12 emptyStack = -1;
13 top = emptyStack;
14 items = new char[maxStack];
15 }
16
Allow us to keep your files
>> generate tests
Home:
EvoSuite
About the tool:
Documentation
Download
Try EvoSuite Online
Publications
Contact:
SE chair at Saarland University
Gordon Fraser
Andrea Arcuri

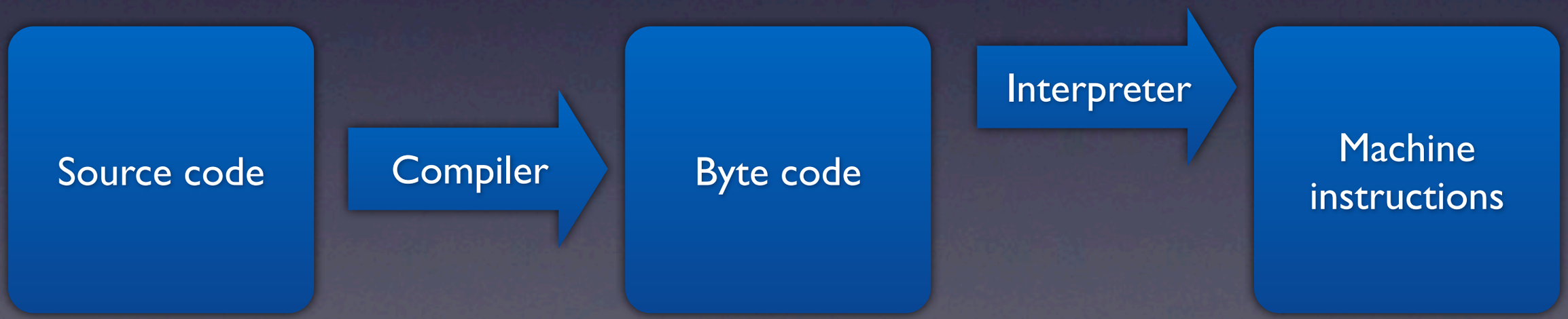
Bytecode

Bytecode


Source code
Compiler
Machine instructions
Source code
Compiler
Byte code

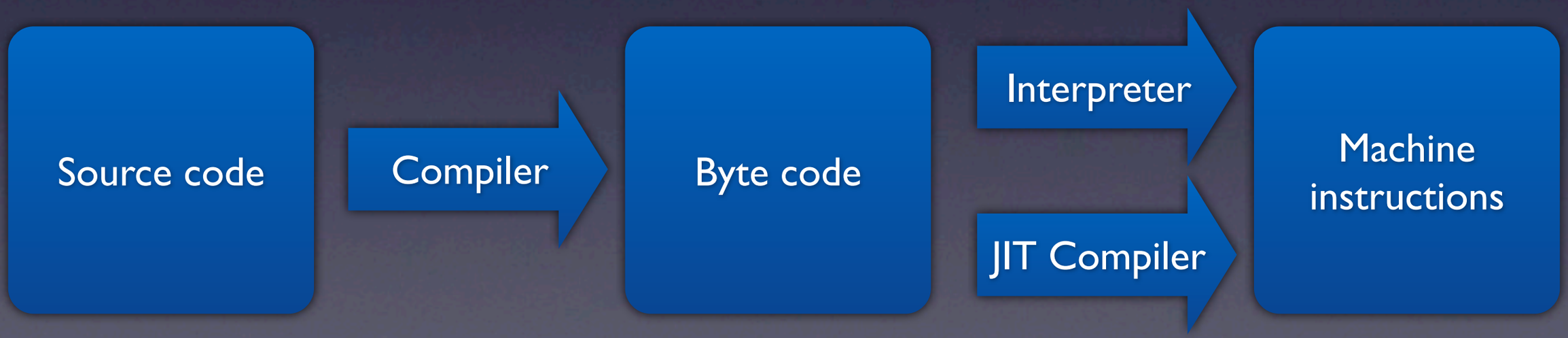
Bytecode


Source code
Compiler
Machine instructions


Source code
Compiler
Byte code
Interpreter
Machine instructions

Bytecode

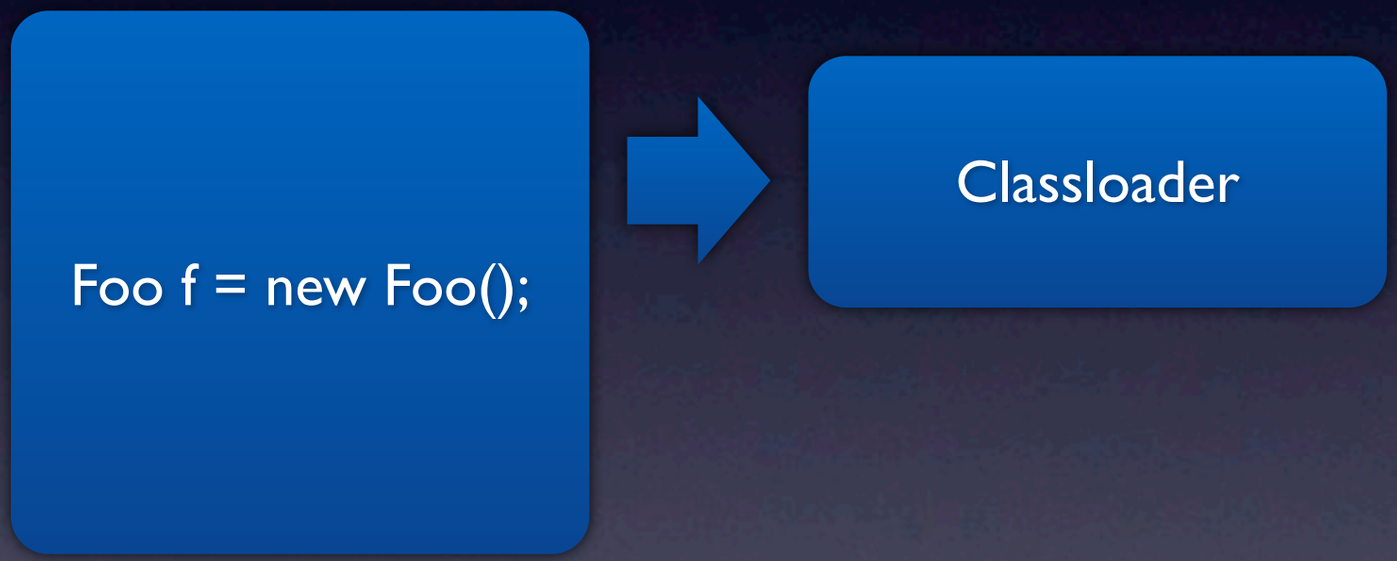

Source code
Compiler
Machine instructions


Source code
Compiler
Byte code
Interpreter
JIT Compiler
Machine instructions

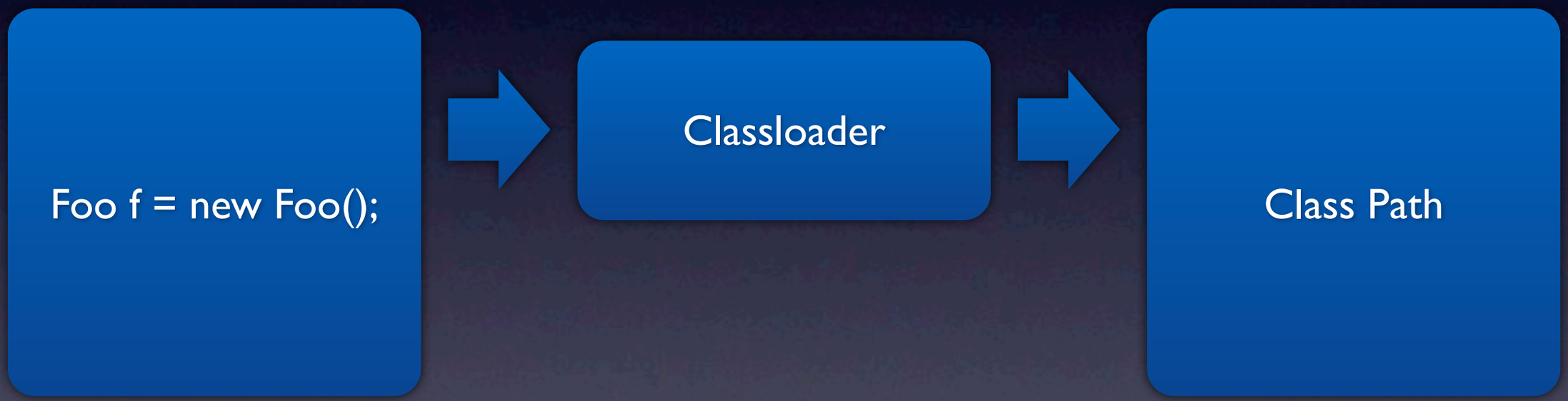
Bytecode Instrumentation

```
Foo f = new Foo();
```

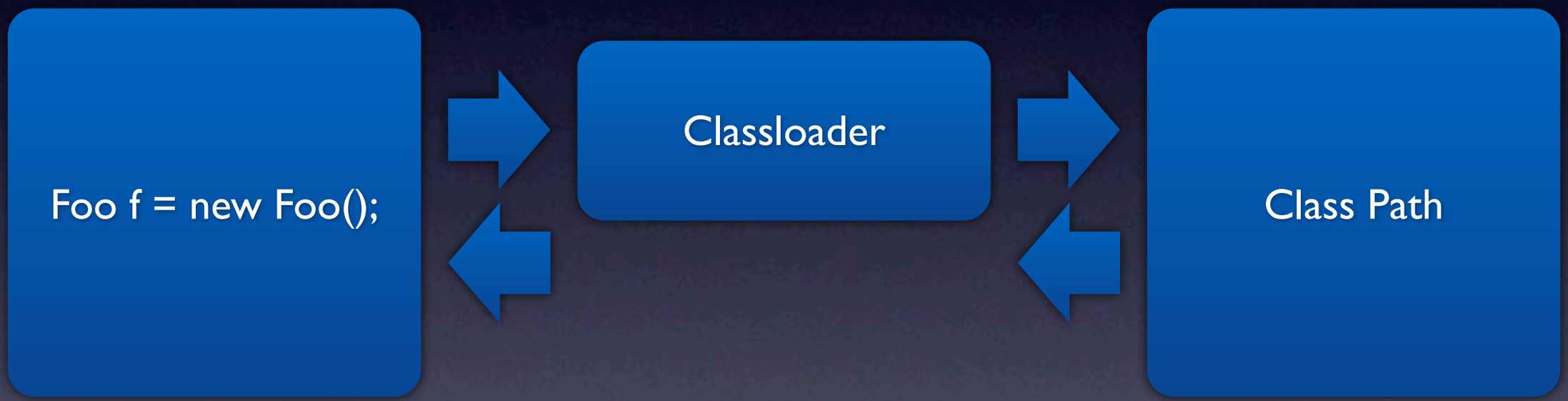
Bytecode Instrumentation



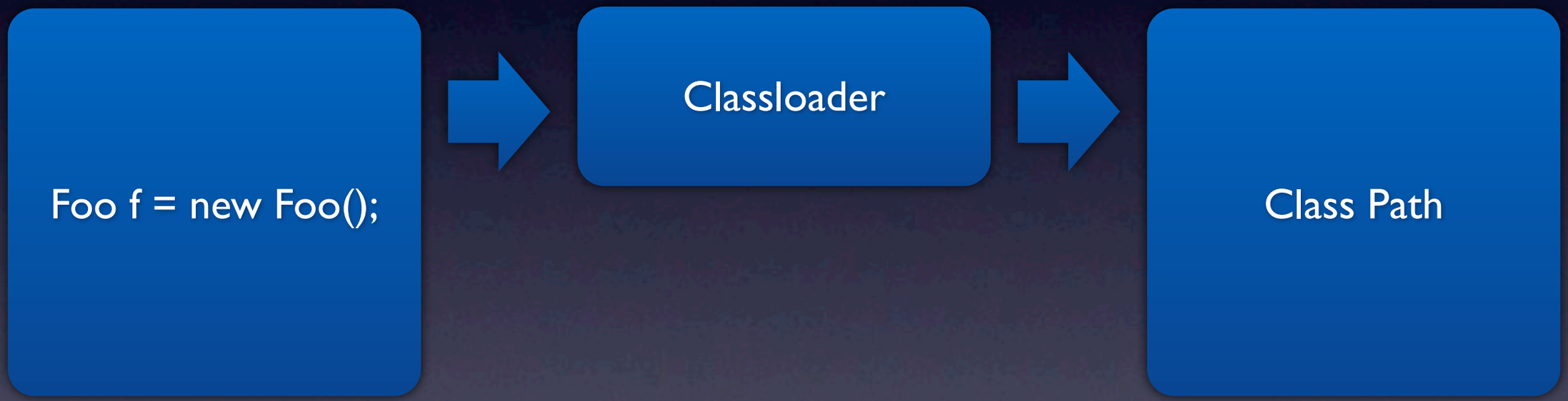
Bytecode Instrumentation



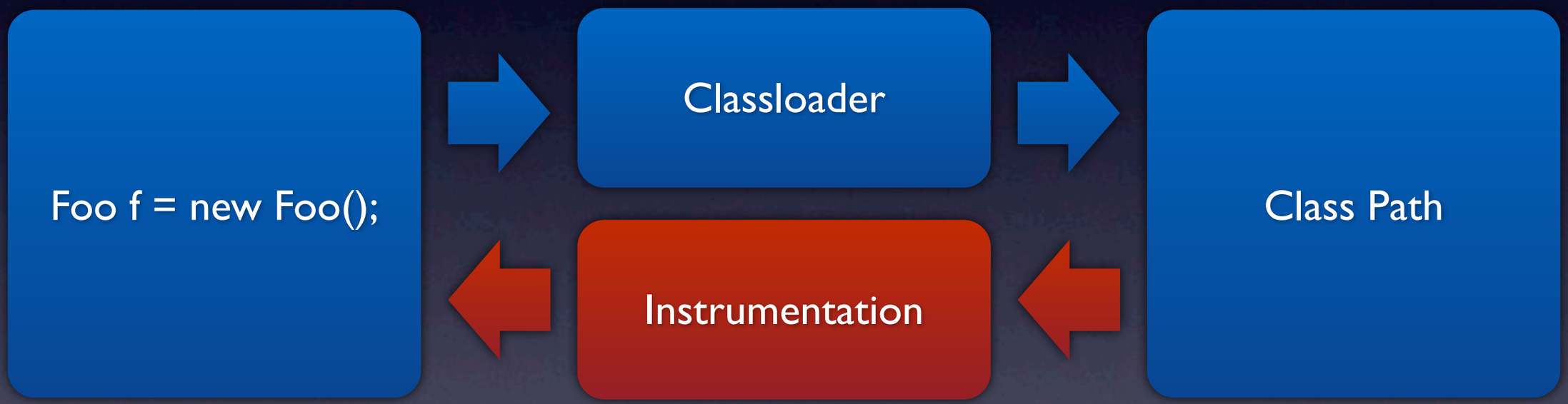
Bytecode Instrumentation



Bytecode Instrumentation



Bytecode Instrumentation



Bytecode

```
ICONST_0
```

Bytecode

```
ICONST_0
```

0

Bytecode

```
ICONST_0
ICONST_1
```

Bytecode

```
ICONST_0
ICONST_1
```

Bytecode

```
ICONST_0
ICONST_1
IADD
```

Bytecode

```
ICONST_0
ICONST_1
IADD
```

Bytecode

```
ICONST_0
ICONST_1
IADD
```

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
```

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
...
```

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
...
L1  INVOKEVIRTUAL ...
```

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
...
INVOKEVIRTUAL ...
```

X

L1

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
...
INVOKEVIRTUAL ...
IRETURN
```

X

L1

Bytecode

```
ICONST_0
ICONST_1
IADD
IFNE L1
...
L1  INVOKEVIRTUAL ...
IRETURN
```

Bytecode Branches

Bytecode Branches

```
if(x == 0)        IFEQ
if(x != 0)        IFNE
```

Bytecode Branches

`if(x == 0)`	`IFEQ`
`if(x != 0)`	`IFNE`
`if(x == y)`	`IF_ICMPEQ`
`if(x <= y)`	`IF_ICMPLT`

Bytecode Branches

`if(x == 0)`	`IFEQ`
`if(x != 0)`	`IFNE`
`if(x == y)`	`IF_ICMPEQ`
`if(x <= y)`	`IF_ICMPLT`
`if(x == null)`	`IFNULL`
`if(x == this)`	`IF_ACMPEQ`

Side Effects

```
if(foo())
  // some code
else
  // else ...
```

Side Effects

```
if(foo())
  // some code
else
  // else ...
```

```
0:  ALOAD_0
1:  INVOKEVIRTUAL foo()Z
2:  IFEQ 5
3:  // else ...
4:  GOTO 6
5:  // some code
6:  ...
```

Bytecode Instrumentation

```
if(x == 0)
```

Bytecode Instrumentation

```
ILOAD_1
IFEQ
```

Bytecode Instrumentation

```
ILOAD_1
IFEQ
```

```
ILOAD_1
DUP
LDC branchId
LDC opcode of IFEQ
INVOKE distance
IFEQ
```

Bytecode Instrumentation

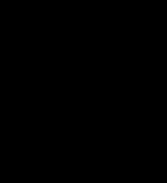
```
if(x >= y)
```

Bytecode Instrumentation

```
ILOAD_1
ILOAD_2
IF_ICMPGT
```

Bytecode Instrumentation

```
ILOAD_1
ILOAD_2
IF_ICMPGT
```



```
ILOAD_1
ILOAD_2
DUP2
LDC branchId
LDC opcode IF_ICMPGT
INVOKE distance
IF_ICMPGT
```

Flag Assignments

```
boolean flag = x <= 0;
```

Flag Assignments

```
boolean flag = x <= 0;
```

```
0:  ILOAD_1
1:  IFLE 4
2:  ICONST_0
3:  GOTO 5
4:  ICONST_1
5:  ISTORE_2
```

Flag Use

```
if(flag)
  // some code
else
  // else code
```

Flag Use

```
if(flag)
  // some code
else
  // else code
```

```
0:  ILOAD_1
1:  IFEQ 4
2:  // some code
3:  GOTO 6
4:  // else code
5:  ...
```

Flag Use

```
if(!flag)
  // some code
else
  // else code
```

Flag Use

```
if(!flag)
  // some code
else
  // else code
```

```
0:  ILOAD_1
1:  IFNE 4
2:  // some code
3:  GOTO 6
4:  // else code
5:  ...
```

Testability Transformation

```
boolean flag = a < b;

if(flag)
  // some code
else
  // else code
```

Testability Transformation

```
boolean flag = a < b;
if(flag)
  // some code
else
  // else code
```

```
int flag = b - a;
if(flag > 0)
  // some code
else
  // else code
```

Bytecode Testability Transformation

```
boolean flag = a > b;
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPLE 5
3:  ICONST_0
4:  GOTO 6
5:  ICONST_1
6:  ISTORE_3
```

Bytecode Testability Transformation

```
boolean flag = a > b;
```

```
int flag = a - b;
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPLE 5
3:  ICONST_0
4:  GOTO 6
5:  ICONST_1
6:  ISTORE_3
```

Bytecode Testability Transformation

```
boolean flag = a > b;
```

```
int flag = a - b;
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPLE 5
3:  ICONST_0
4:  GOTO 6
5:  ICONST_1
6:  ISTORE_3
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  ISUB
3:  ISTORE_3
```

Bytecode Testability Transformation

```
if(flag)
  // some code
else
  // else code
```

```
 6:  ILOAD_3
 7:  IFNE 10
 8:  // some code
 9:  GOTO 11
10:  // else code
```

Bytecode Testability Transformation

```
if(flag)
  // some code
else
  // else code
```

```
if(flag > 0)
  // some code
else
  // else code
```

```
 6:  ILOAD_3
 7:  IFNE 10
 8:  // some code
 9:  GOTO 11
10:  // else code
```

Bytecode Testability Transformation

```
if(flag)
  // some code
else
  // else code
```

```
 6:  ILOAD_3
 7:  IFNE 10
 8:  // some code
 9:  GOTO 11
10:  // else code
```

```
if(flag > 0)
  // some code
else
  // else code
```

```
 6:  ILOAD_3
 7:  IFLT 10
 8:  // some code
 9:  GOTO 11
10:  // else code
```

Function Assigned Flags

```
boolean foo(int a, int b) {
   if(a > b)
      return true;
   else
      return false;
}
```

Function Assigned Flags

```
boolean foo(int a, int b) {
   if(a > b)
      return true;
   else
      return false;
}
```

```
0:   ILOAD_1
1:   ILOAD_2
2:   IF_ICMPLE 5
3:   ICONST_1
4:   IRETURN
5:   ICONST_0
6:   IRETURN
```

Function Assigned Flags

```
int foo(int a, int b) {
  if(a > b)
    return a - b;
  else
    return a - b;
}
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  ISUB
3:  IRETURN
```

Function Assigned Flags

```
boolean foo(int a, int b) {
   if(a > b) {
      // ...
      return true;
   } else {
      // ...
      return false;
   }
}
```

Function Assigned Flags

```
boolean foo(int a, int b) {
  if(a > b) {
    // ...
    return true;
  } else {
    // ...
    return false;
  }
}
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPLE 5
// ...
3:  ICONST_1
4:  IRETURN
// ...
5:  ICONST_0
6:  IRETURN
```

Bytecode Testability Transformation

```
int foo(int a, int b) {
  int tmp = |a - b|;
  if(a < b) {
    // ...
    return tmp;
  } else {
    // ...
    return -tmp;
  }
}
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  DUP2
3:  ISUB
4:  INVOKE Math.abs
5:  ISTORE_3
6:  IF_ICMPGE 9
```

Bytecode Testability Transformation

```
int foo(int a, int b) {
  int tmp = |a - b|;
  if(a < b) {
    // ...
    return tmp;
  } else {
    // ...
    return -tmp;
  }
}
```

```
 6:  IF_ICMPGE 9
     // ...
 7:  ILOAD_3
 8:  IRETURN
 9:  // ...
10:  ILOAD_3
11:  INEG
12:  IRETURN
```

Bytecode Testability Transformation

IFEQ, IFNE, IFLT,
IFGT, IFLE, IFGE

IF_ICMPEQ, IF_ICMPNE,
IF_ICMPLT, IF_ICMPGT,
IF_ICMPLE, IF_ICMPGE

Bytecode Testability Transformation

IFEQ, IFNE, IFLT,
IFGT, IFLE, IFGE

IF_ICMPEQ, IF_ICMPNE,
IF_ICMPLT, IF_ICMPGT,
IF_ICMPLE, IF_ICMPGE

```
DUP
INVOKE Math.abs
LDC K
IADD
```

Bytecode Testability Transformation

IFEQ, IFNE, IFLT,
IFGT, IFLE, IFGE

```
DUP
INVOKE Math.abs
LDC K
IADD
```

IF_ICMPEQ, IF_ICMPNE,
IF_ICMPLT, IF_ICMPGT,
IF_ICMPLE, IF_ICMPGE

```
DUP2
ISUB
INVOKE Math.abs
LDC K
IADD
```

Compound Predicates

```
if(a == b && a > 0)
   return true;
else
   return false;
```

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPNE 7
3:  ILOAD_1
4:  IFLE 7
5:  ICONST_1
6:  IRETURN
7:  ICONST_0
8:  IRETURN
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
2:  IF_ICMPNE 7
3:  ILOAD_1
4:  IFLE 7
5:  ICONST_1
6:  IRETURN
7:  ICONST_0
8:  IRETURN
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
4:  IFLE 7
5:  ICONST_1
6:  IRETURN
7:  ICONST_0
8:  IRETURN
```

Compound Predicates

```
0:   ILOAD_1
1:   ILOAD_2
     DUP2
     ISUB
     INVOKE distance
2:   IF_ICMPNE 7
3:   ILOAD_1
     DUP
     INVOKE distance
4:   IFLE 7
5:   ICONST_1
6:   IRETURN
7:   ICONST_0
8:   IRETURN
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
    DUP
    INVOKE distance
4:  IFLE 7
5:  ICONST_1
    INVOKE getDistance
6:  IRETURN
7:  ICONST_0
8:  IRETURN
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
    DUP
    INVOKE distance
4:  IFLE 7
5:  ICONST_1
    INVOKE getDistance
6:  IRETURN
7:  ICONST_0
    INVOKE getDistance
8:  IRETURN
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
    DUP
    INVOKE distance
4:  IFLE 7
5:  ICONST_1
    INVOKE getDistance
6:  IRETURN
7:  ICONST_0
    INVOKE getDistance
8:  IRETURN
```

```
int getDistance(boolean sign) {
  if(sign)
    return 1+D/2^L;
  else
    return -(1+D)/2^L;
}
```

Bytecode Flags

```
if(x == 1.0)
  // some code
else
  // else code
```

Bytecode Flags

```
if(x == 1.0)
  // some code
else
  // else code
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```

Bytecode Flags

If the two numbers are the same, the 32-bit integer 0 is pushed onto the stack. If value2 is greater than value1, the integer 1 is pushed onto the stack. If value1 is greater than value2, -1 is pushed onto the stack. If either numbers is NaN, the integer 1 is pushed onto the stack.

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```

Bytecode Flags

```
if(x == 1.0)
  // some code
else
  // else code
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```

Bytecode Flags

```
if(x == 1.0)
  // some code
else
  // else code
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DSUB
3:  INVOKE distance
4:  IFEQ 6
5:  // some code
6:  GOTO 7
7:  // else code
```

Bytecode Flags

```
if(x == 1.0)
  // some code
else
  // else code
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```

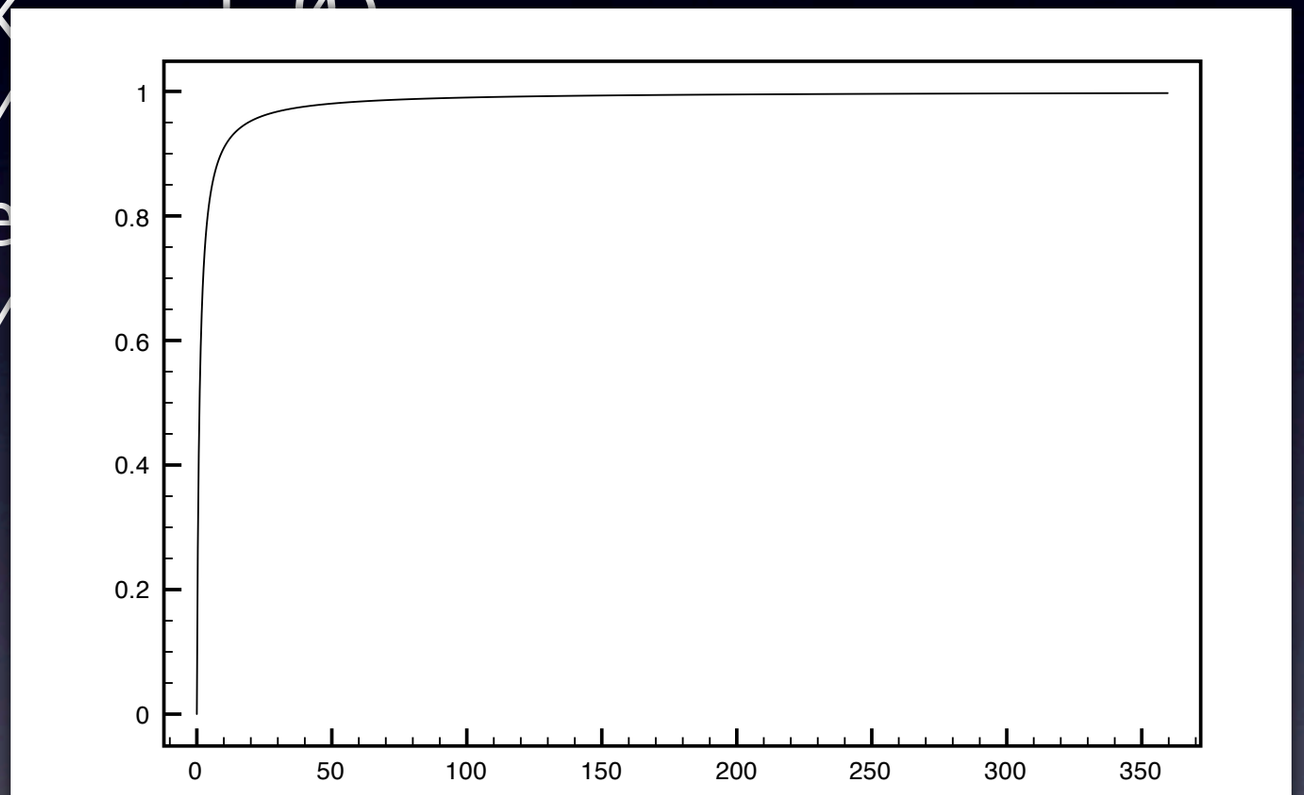
```
0:  DLOAD_1
1:  LDC 1.0
2:  DSUB
3:  INVOKE distance
4:  IFEQ 6
```

```
int distance(d) {
  return K * sign(d) * abs(d) / (abs(d) + 1);
}
```

Bytecode Flags

```
if(x
  //
else
  //
```

```
0:  DLOAD_1
1:  LDC 1.0
2:  DCMPL
3:  IFEQ 6
4:  // some code
5:  GOTO 7
6:  // else code
7:  ...
```




```
int distance(d) {
  return K * sign(d) * abs(d) / (abs(d) + 1);
}
```

Signature Transformation

```
class Foo {
   boolean x;

   boolean bar(boolean y) {
      ...
   }
}
```

```
class Foo {
   int x;

   int bar(int y) {
      ...
   }
}
```

Signature Transformation

```
Foo foo = new Foo();
boolean x = foo.bar(true);
```

```
INVOKESPECIAL Foo:<init>()
ASTORE_1
ALOAD_1
INVOKEVIRTUAL bar(Z)Z
```

```
Foo foo = new Foo();
int x = foo.bar(1);
```

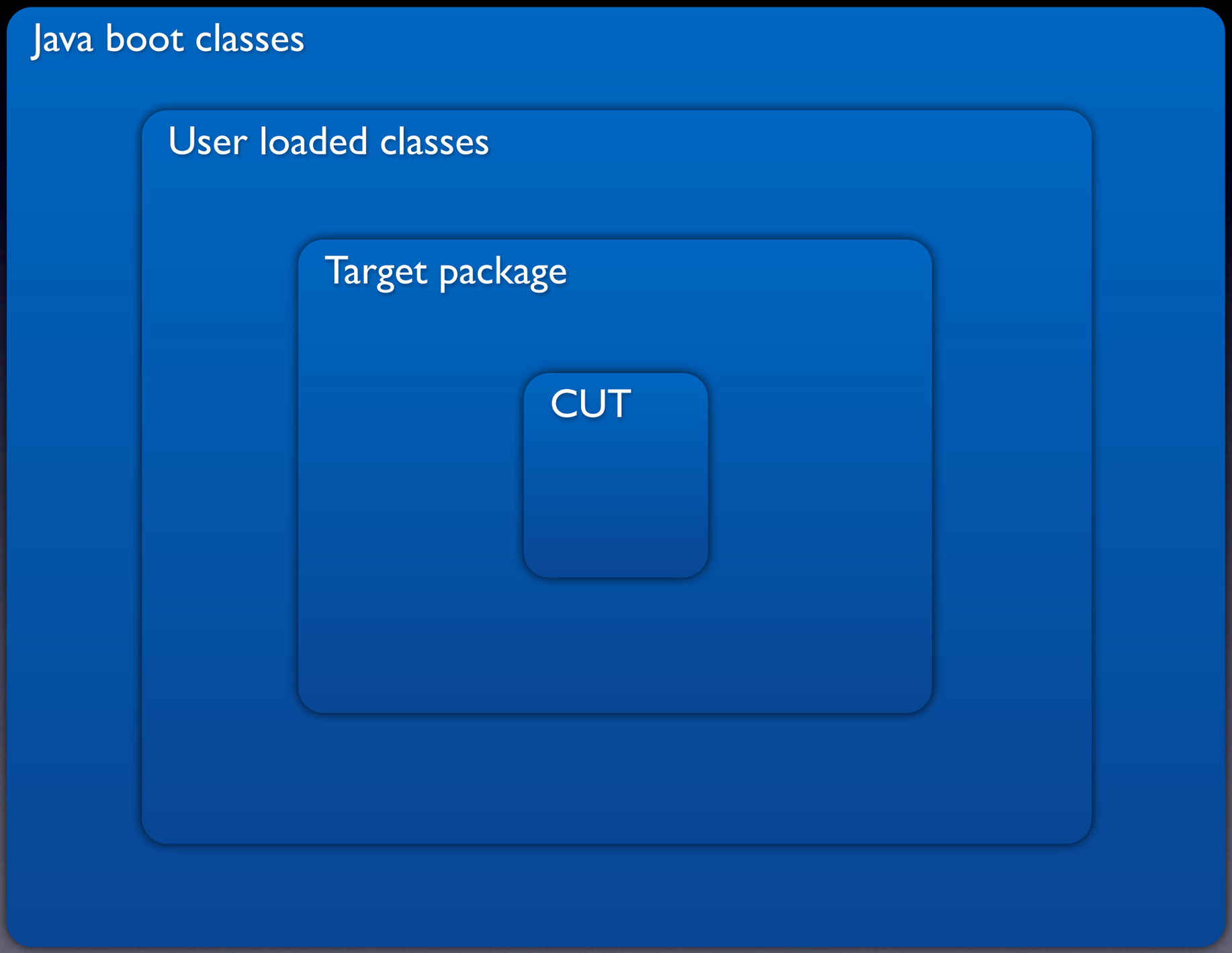
```
INVOKESPECIAL Foo:<init>()
ASTORE_1
ALOAD_1
INVOKEVIRTUAL bar(I)I
```

Signature Transformation

```
Foo foo1 = new Foo();
Foo foo2 = new Foo();
boolean x = foo1.equals(foo2);
```

```
INVOKESPECIAL Foo:<init>()
ASTORE_1
INVOKESPECIAL Foo:<init>()
ASTORE_2
ALOAD_1
ALOAD_2
INVOKEVIRTUAL Object:equals(Object)Z
```

Transformation Boundaries



Transformation Boundaries

Transformation Boundaries

- Integer to {0, 1} for outgoing boolean parameters

Transformation Boundaries

- Integer to {0, 1} for outgoing boolean parameters
- 0/1 to -K/+K for incoming boolean values

Transformation Boundaries

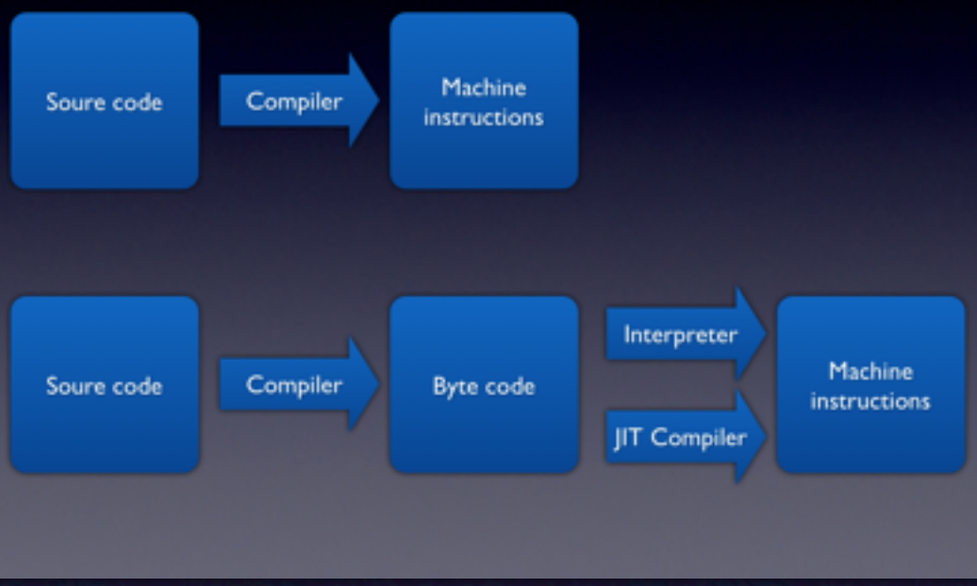
- Integer to {0,1} for outgoing boolean parameters
- 0/1 to -K/+K for incoming boolean values
- Duplicate methods:

```
boolean flagMethod() {...}
int __flagMethod() {...}
```

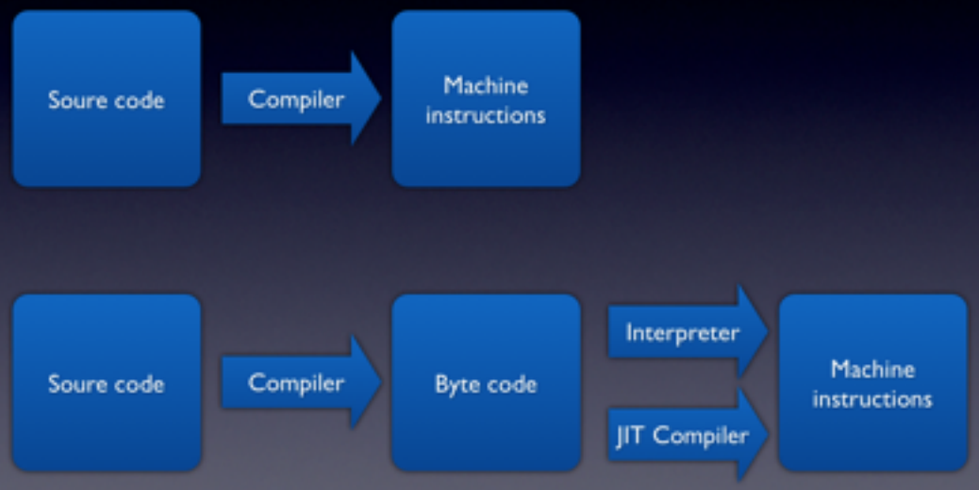
Evaluation

Library	Classes	Improvement
Commons Codec	21	6
Commons CLI	14	5
Java Collections	16	11
JDom	18	6

Bytecode


Soure code
Compiler
Machine instructions
Soure code
Compiler
Byte code
Interpreter
JIT Compiler
Machine instructions

Bytecode

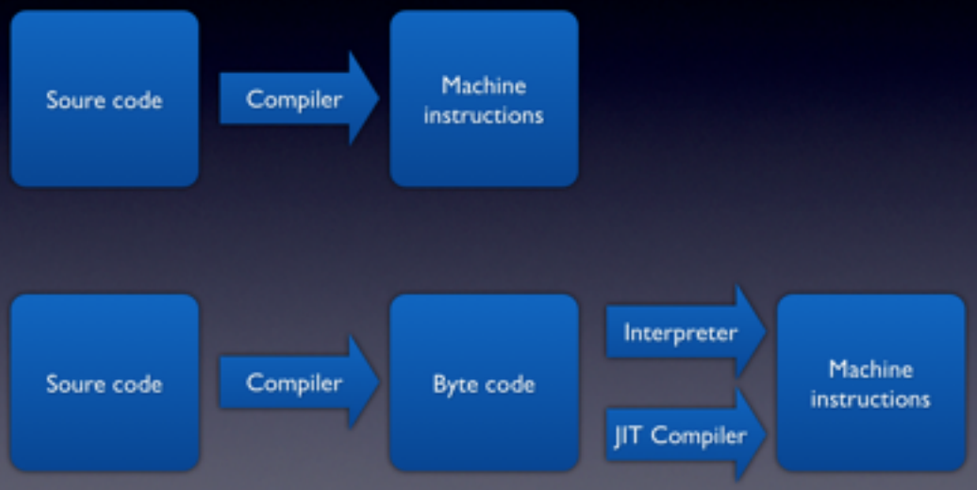


Testability Transformation

```
boolean flag = a < b;

if(flag)
  // some code
else
  // else code
```

```
int flag = b - a;

if(flag > 0)
  // some code
else
  // else code
```

Bytecode



Testability Transformation

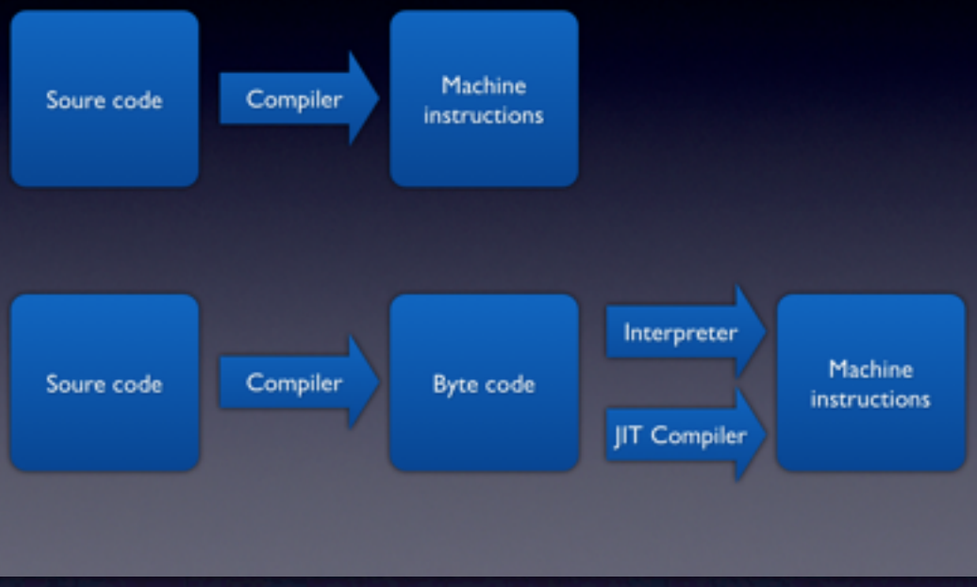
```
boolean flag = a < b;

if(flag)
  // some code
else
  // else code
```

```
int flag = b - a;

if(flag > 0)
  // some code
else
  // else code
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
    DUP
    INVOKE distance
4:  IFLE 7
5:  ICONST_1
    INVOKE getDistance
6:  IRETURN
7:  ICONST_0
    INVOKE getDistance
8:  IRETURN
```

Bytecode



Testability Transformation

```
boolean flag = a < b;

if(flag)
  // some code
else
  // else code
```

```
int flag = b - a;

if(flag > 0)
  // some code
else
  // else code
```

Compound Predicates

```
0:  ILOAD_1
1:  ILOAD_2
    DUP2
    ISUB
    INVOKE distance
2:  IF_ICMPNE 7
3:  ILOAD_1
    DUP
    INVOKE distance
4:  IFLE 7
5:  ICONST_1
    INVOKE getDistance
6:  IRETURN
7:  ICONST_0
    INVOKE getDistance
8:  IRETURN
```

Evaluation

Library	Classes	Improvement
Commons Codec	21	6
Commons CLI	14	5
Java Collections	16	11
JDom	18	6