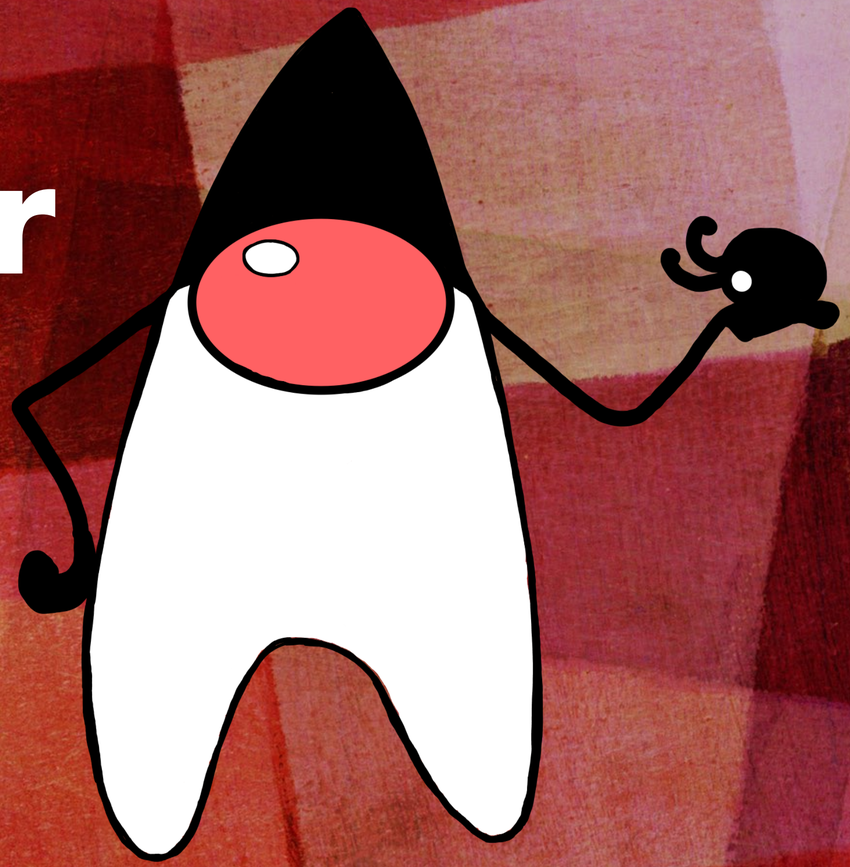
Internal life of your debugger

Nataliia Dziubenko



Nataliia Dziubenko

Software Engineer

Xebia, XSD

@worth_exploring

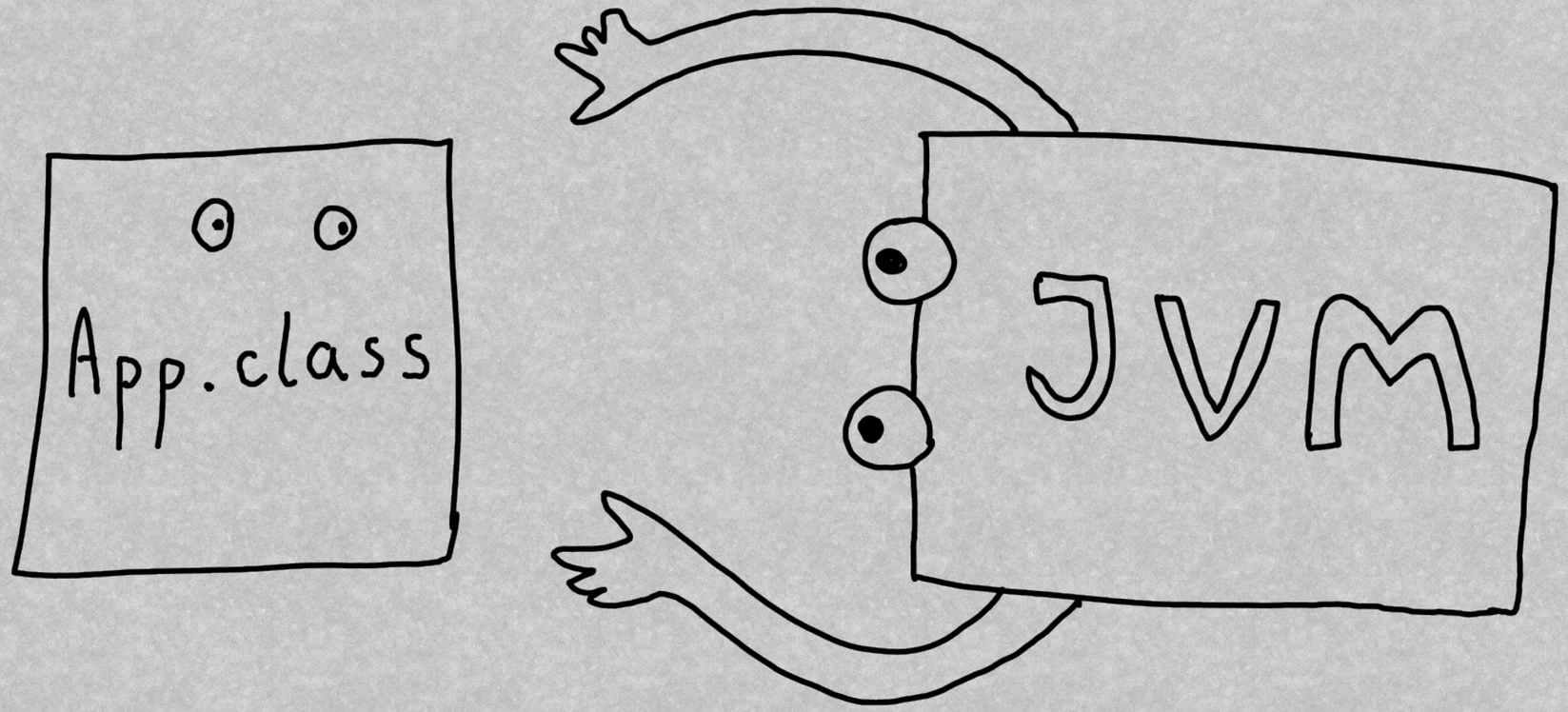
```
...
void do() {
  int a = 5;
  double b = 6d;
  if (a > b){
    System.out.pr...
  } else {
    a = 0;
    System.out ..
  }
  ..
```

App.java

COMPILE

```
0: CA FE BA BE
10: 63 6F 6D 2F
20: 69 6E 67 70
30: 62 74 6F 2F
40: 63 65 42 54
50: 6C 61 6E 67
...
```

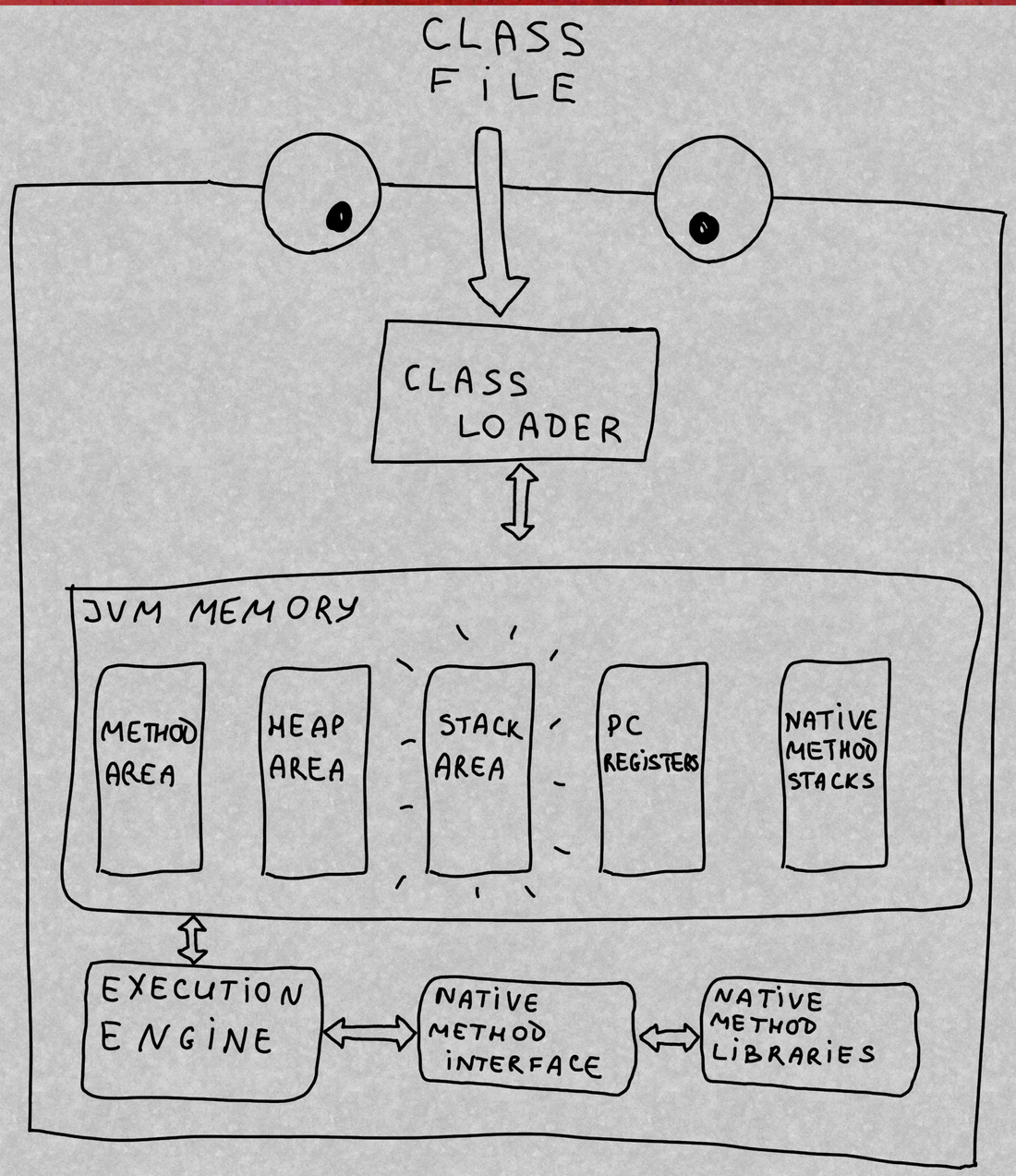
App.class

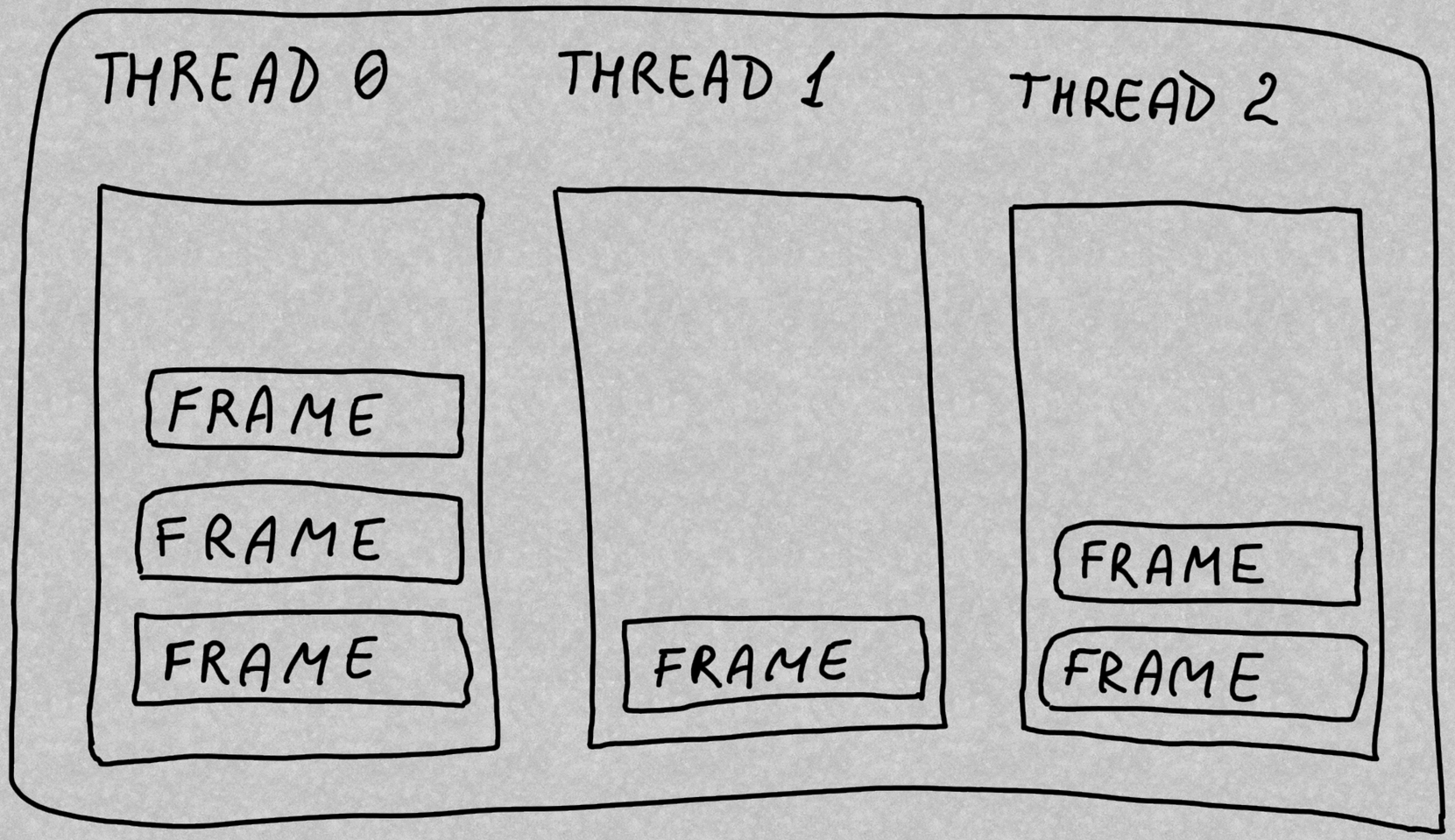

App.class
JVM

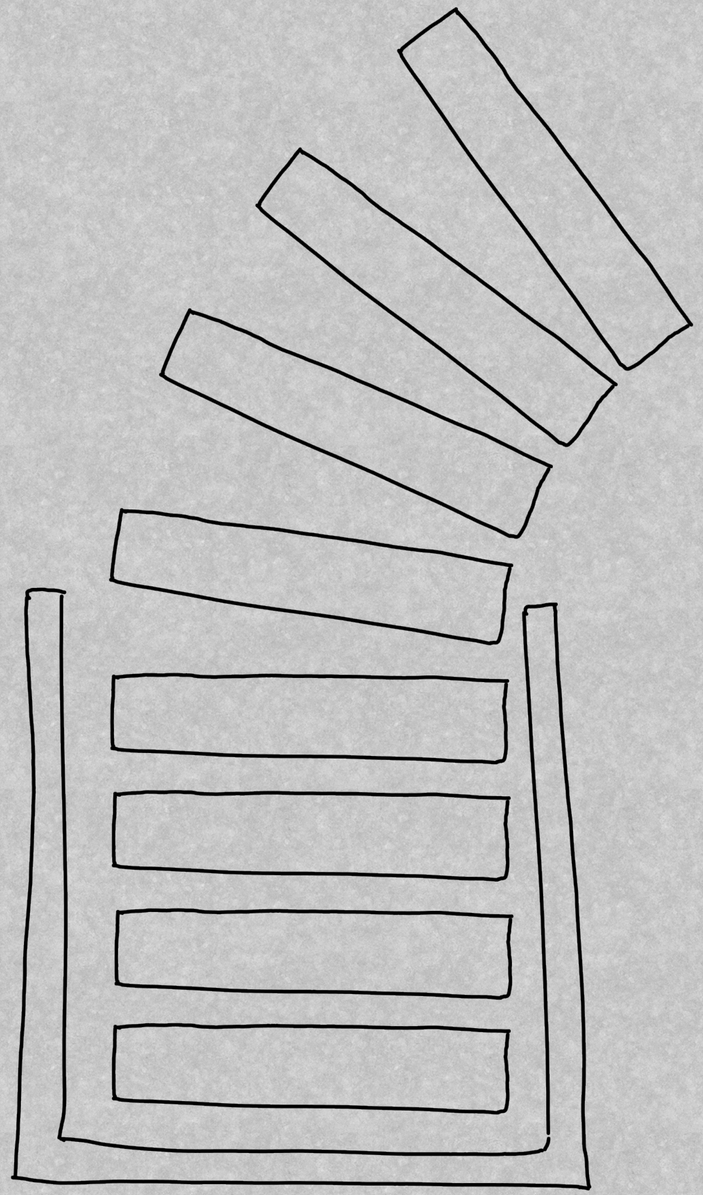
```
javap -c Test
```

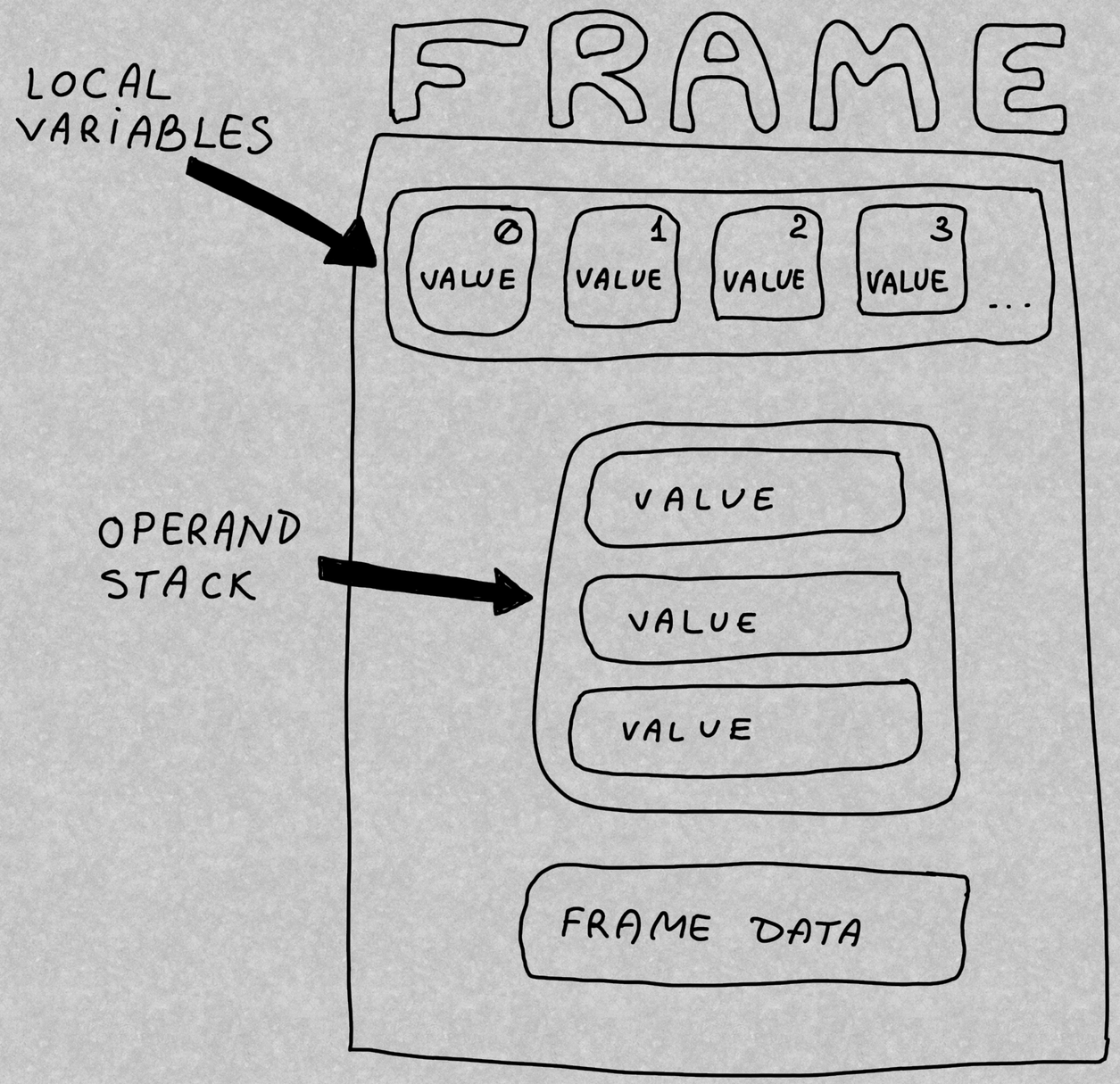
```java
int a = 0;
a += 88;
System.out.print(a);
a--;
a = a * a;
String test = a + "test";
System.out.println(test + " hello");
```

```
Code:
   0: iconst_0
   1: istore_1
   2: iinc          1, 88
   5: getstatic     #2
   8: iload_1
   9: invokevirtual #3
  12: iinc          1, -1
  15: iload_1
  16: iload_1
  17: imul
  18: istore_1
  19: iload_1
  20: invokedynamic #4,  0
  25: astore_2
  26: getstatic     #2
  29: aload_2
  30: invokedynamic #5,  0
  35: invokevirtual #6
  38: return
```

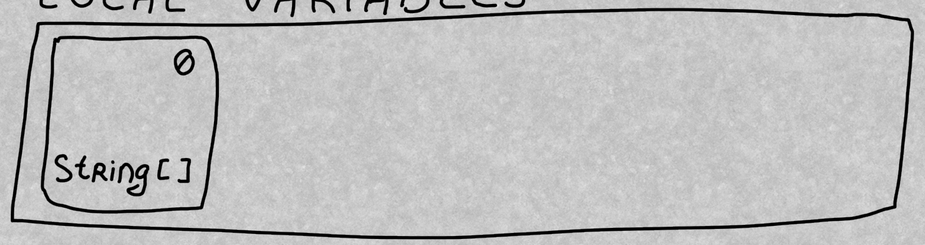

CLASS FILE
CLASS LOADER
JVM MEMORY
METHOD AREA
HEAP AREA
STACK AREA
PC REGISTERS
NATIVE METHOD STACKS
EXECUTION ENGINE
NATIVE METHOD INTERFACE
NATIVE METHOD LIBRARIES


THREAD 0
THREAD 1
THREAD 2
FRAME
FRAME
FRAME
FRAME
FRAME
FRAME

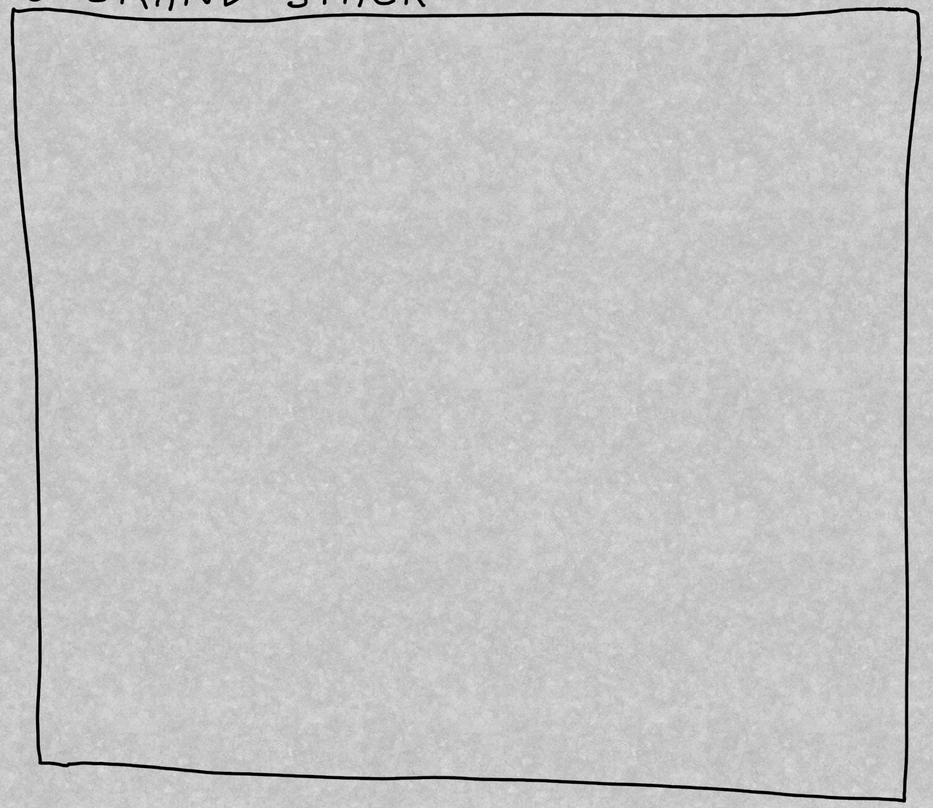


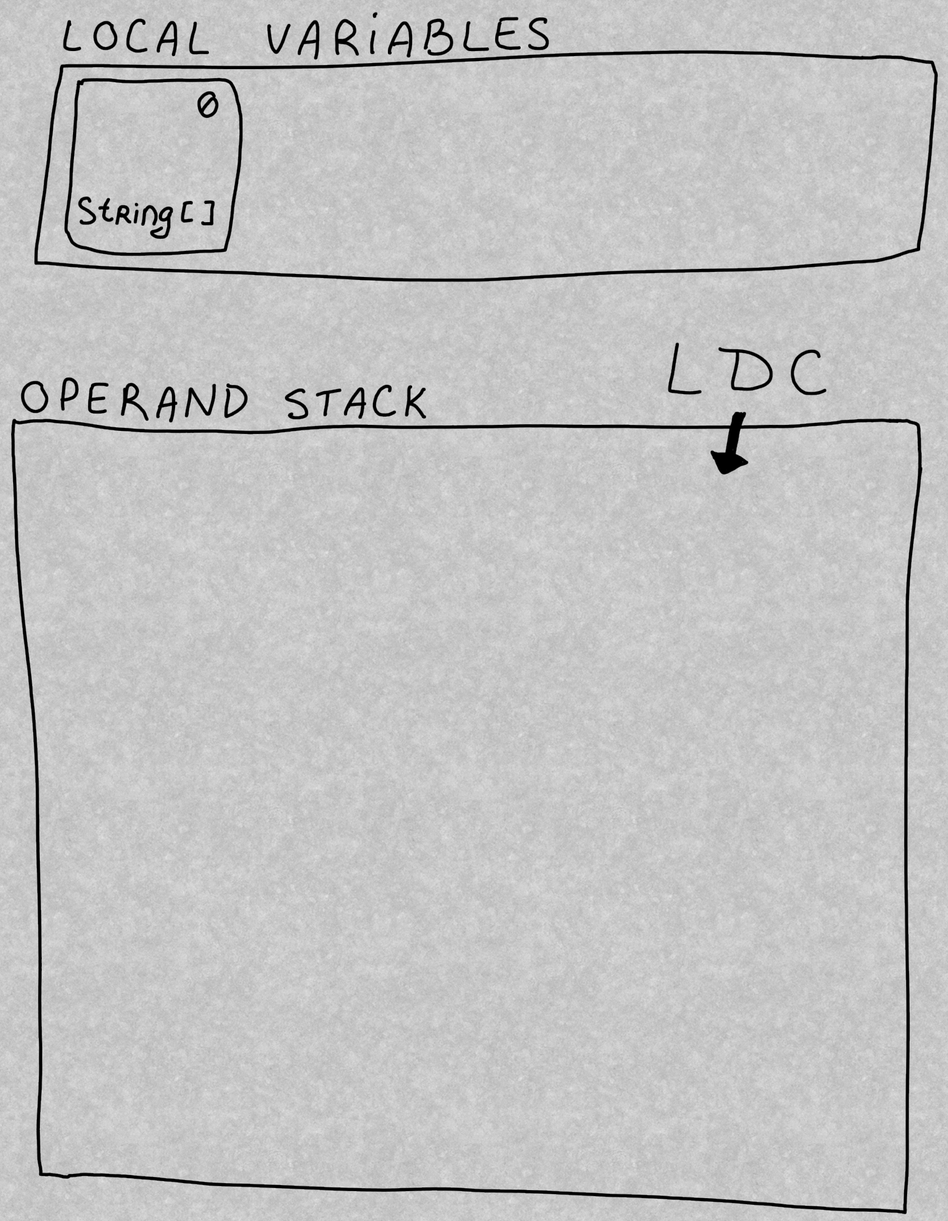

FRAME
LOCAL VARIABLES
0
VALUE
1
VALUE
2
VALUE
3
VALUE
...
OPERAND STACK
VALUE
VALUE
VALUE
FRAME DATA

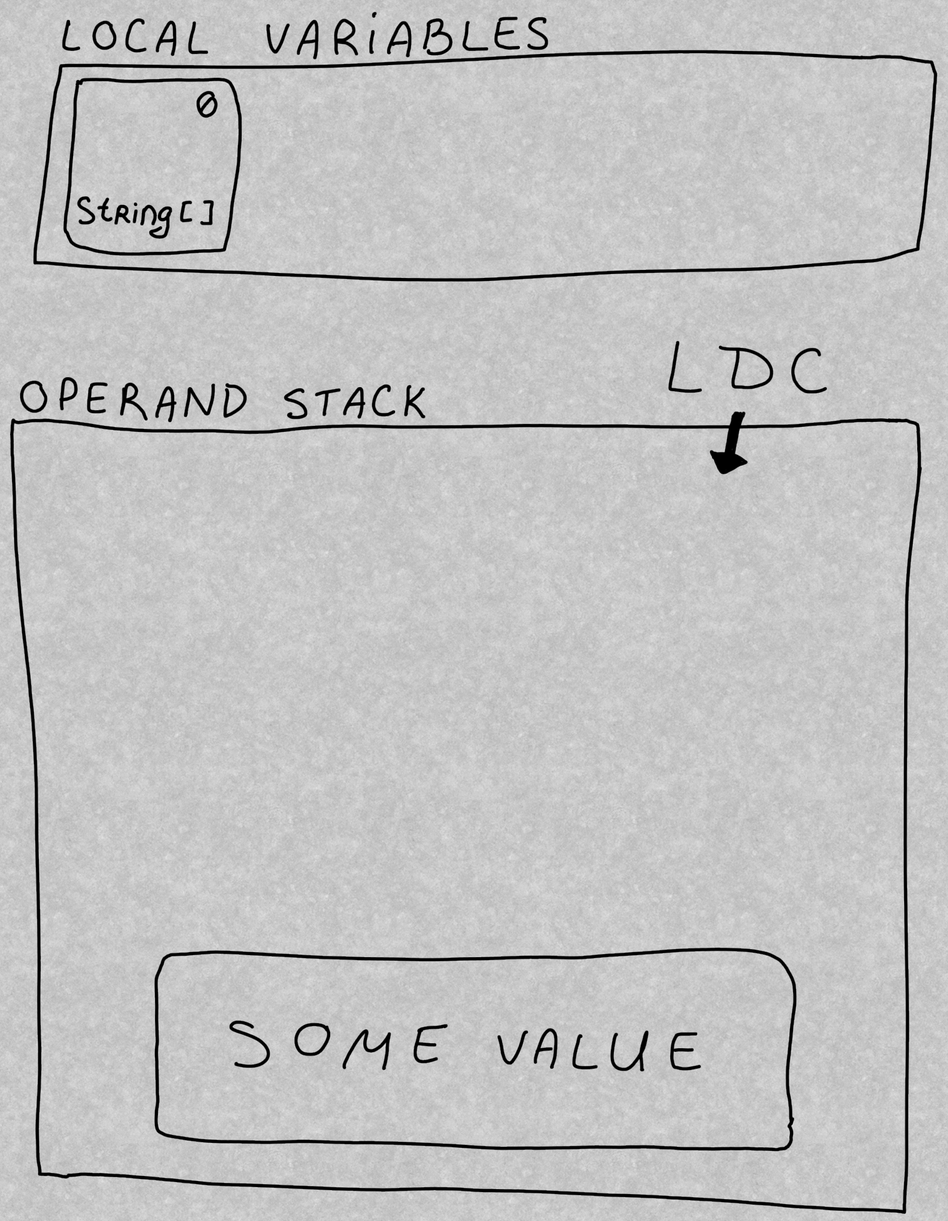
LOCAL VARIABLES

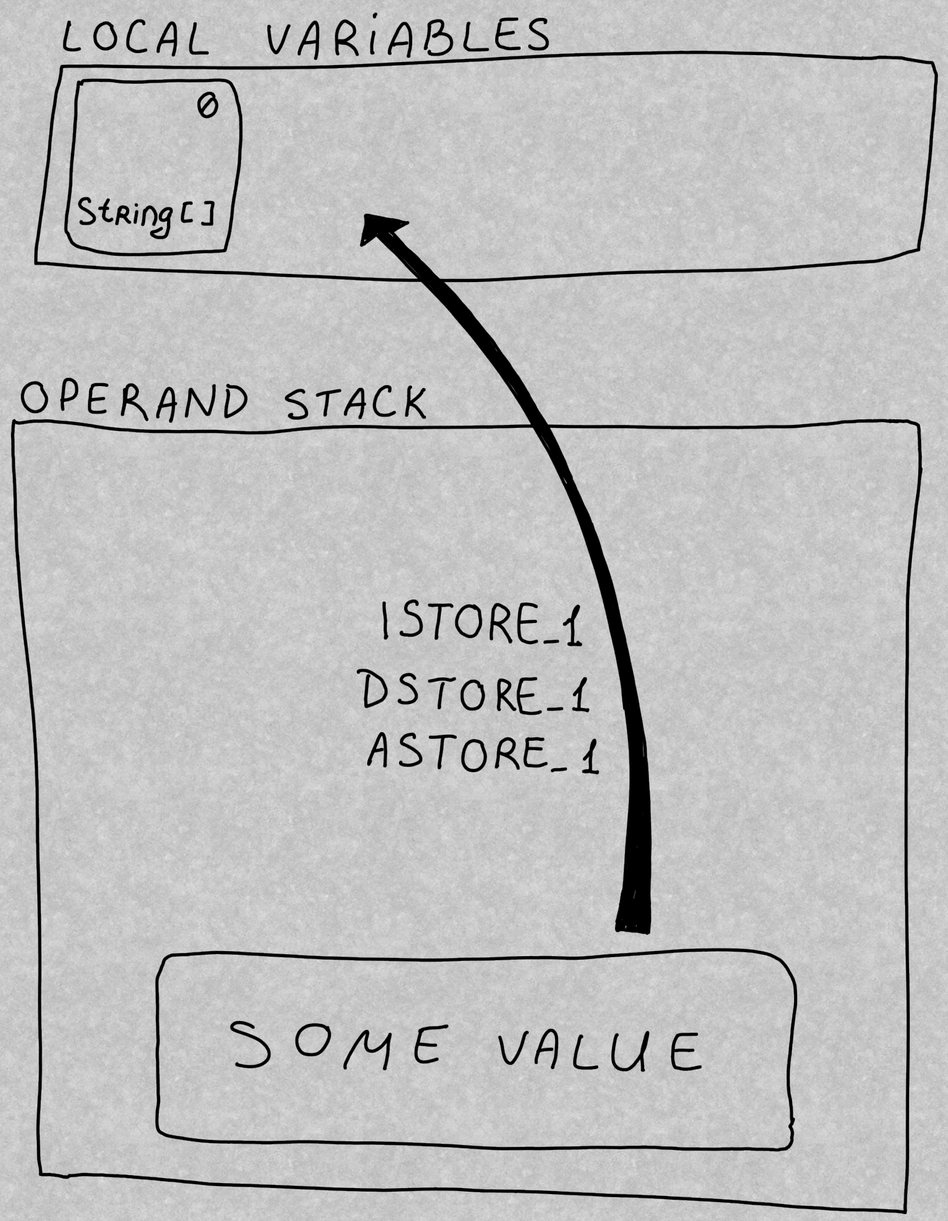


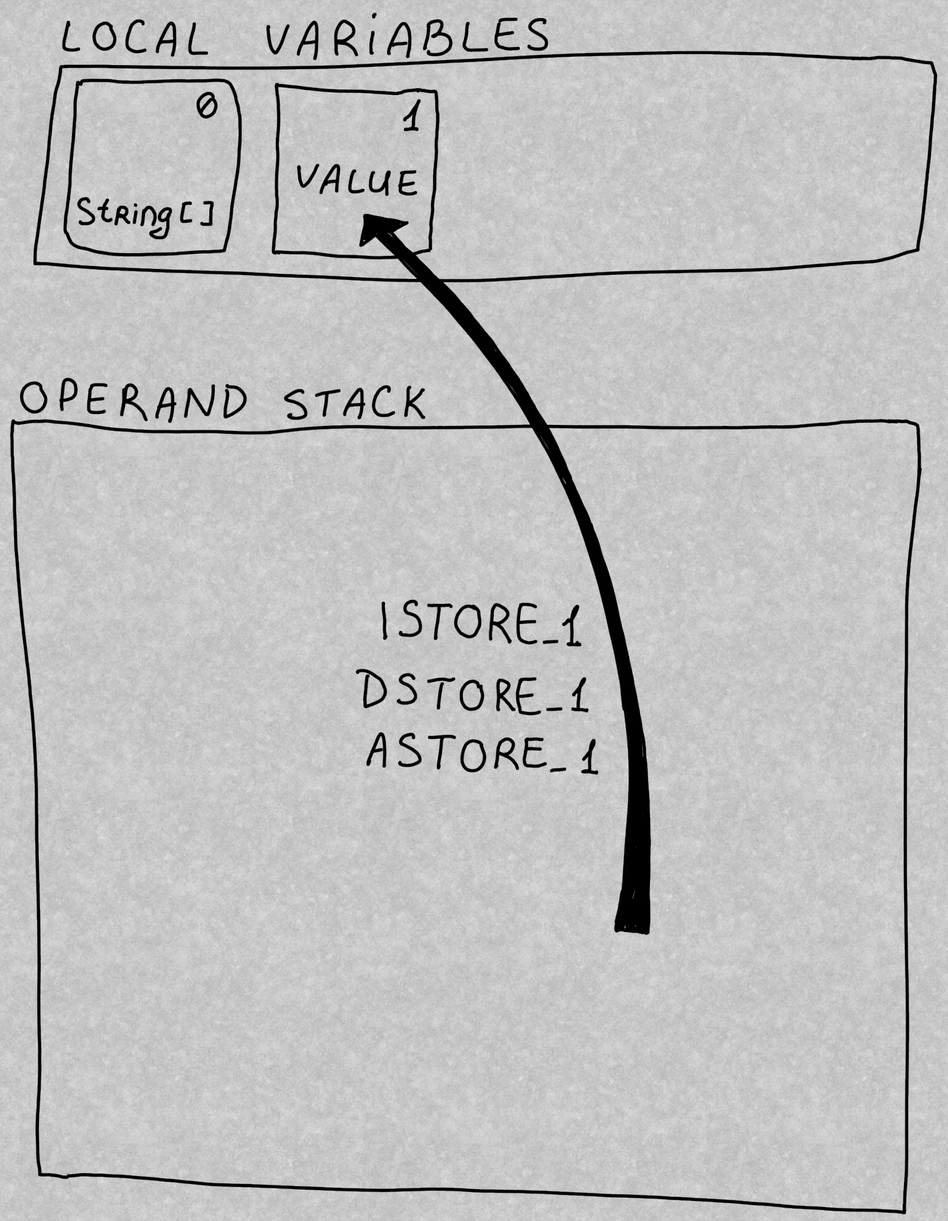
OPERAND STACK

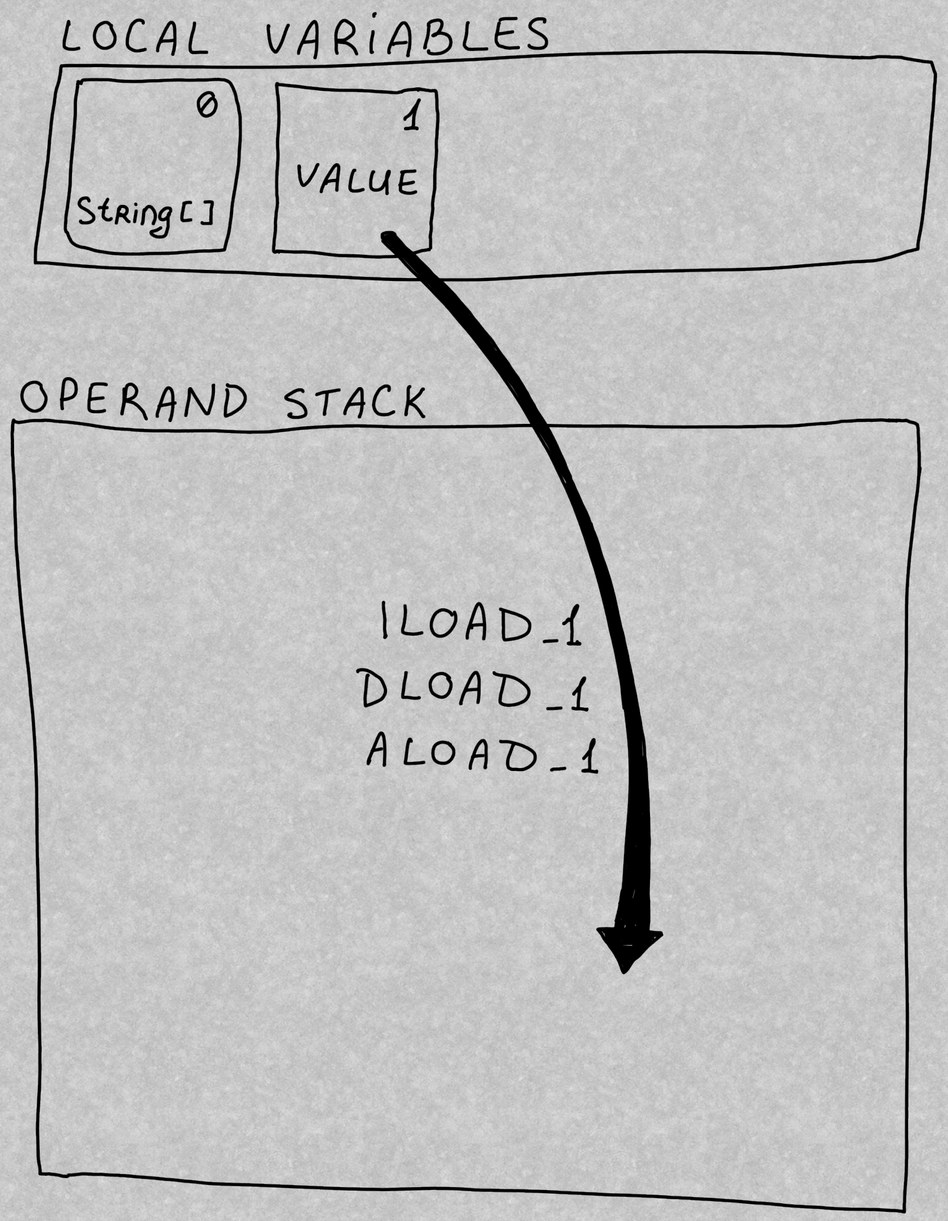


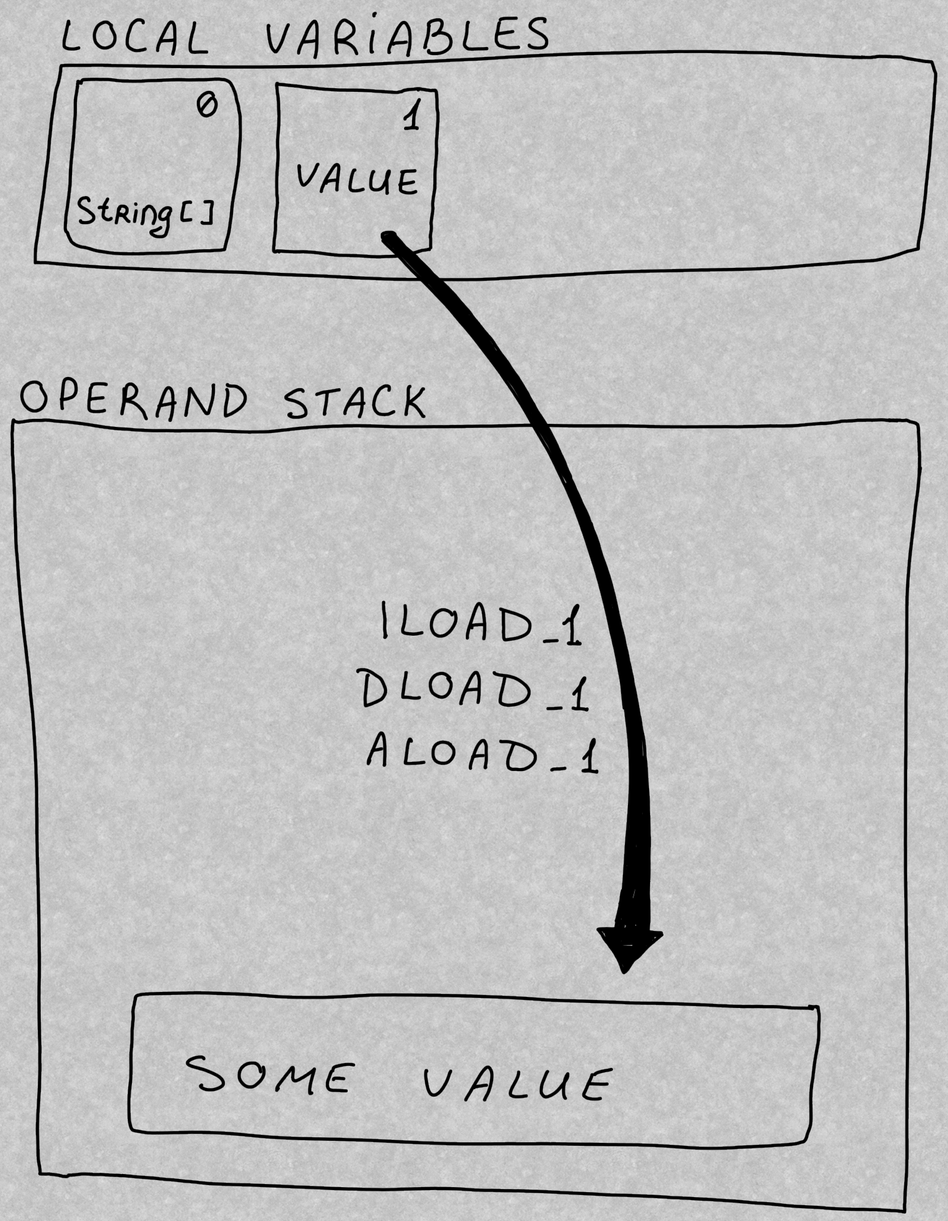

LOCAL VARIABLES
0
String[]
OPERAND STACK
LDC

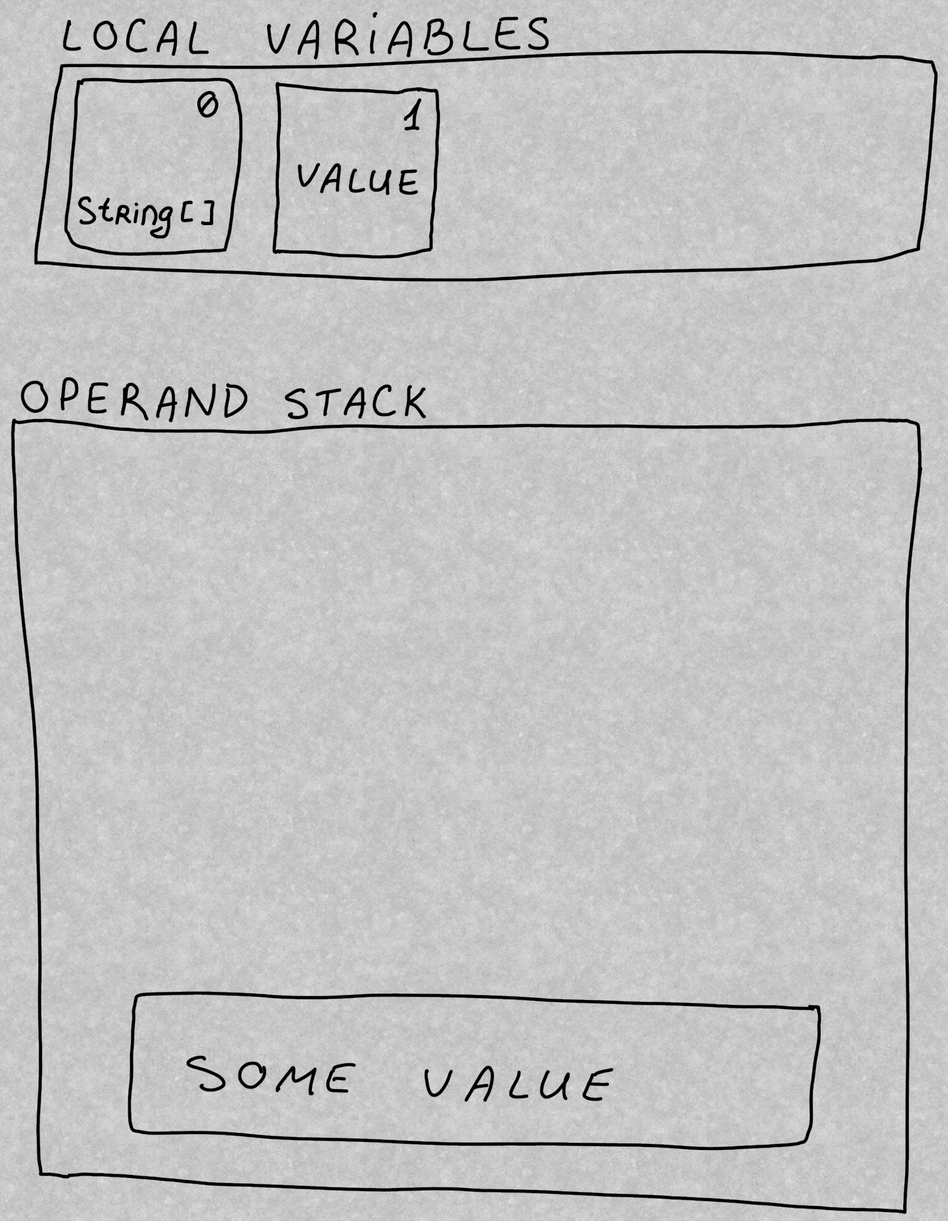

LOCAL VARIABLES
0
String[]
OPERAND STACK
LDC
SOME VALUE

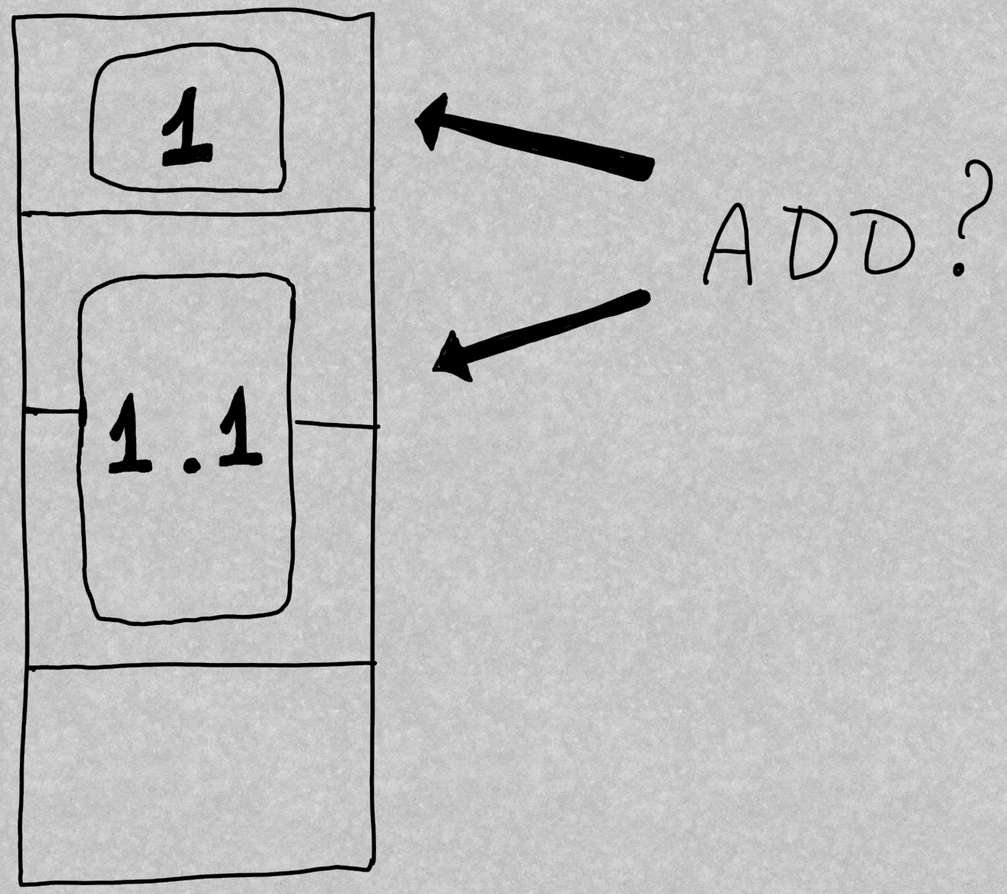

LOCAL VARIABLES
0
String[]
OPERAND STACK
ISTORE_1
DSTORE_1
ASTORE_1
SOME VALUE

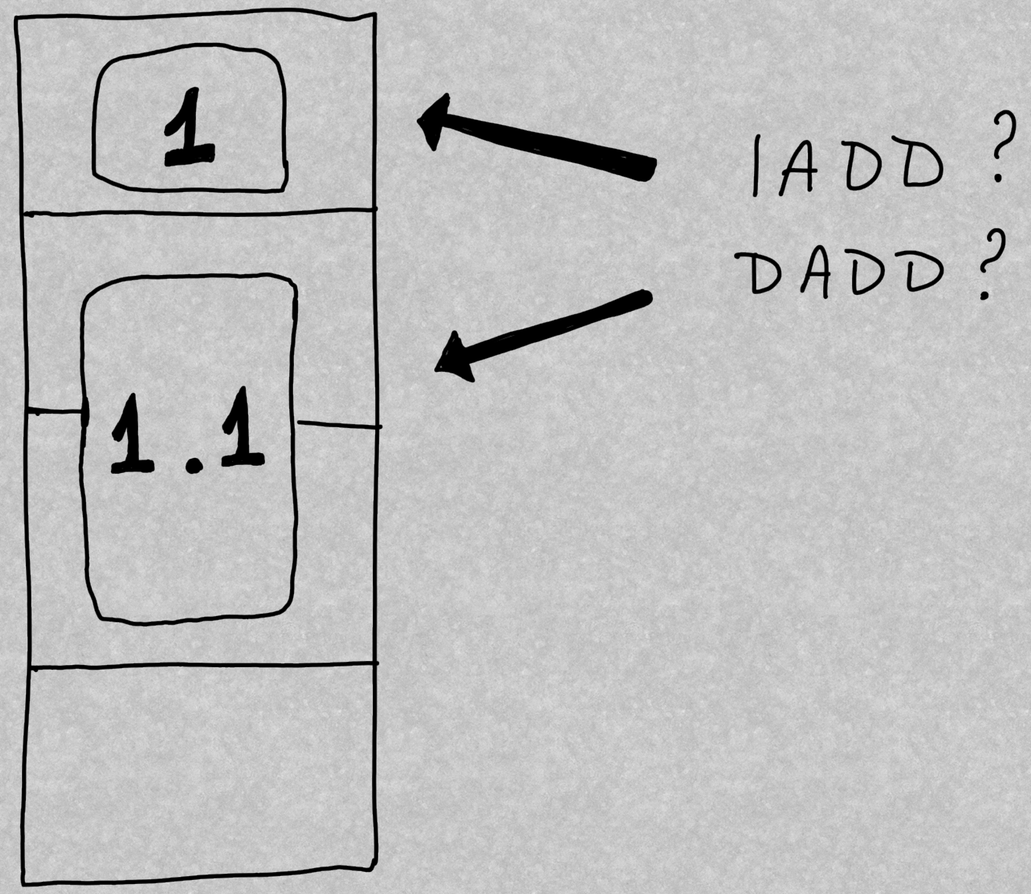

LOCAL VARIABLES
0
String[]
1
VALUE
OPERAND STACK
ISTORE_1
DSTORE_1
ASTORE_1

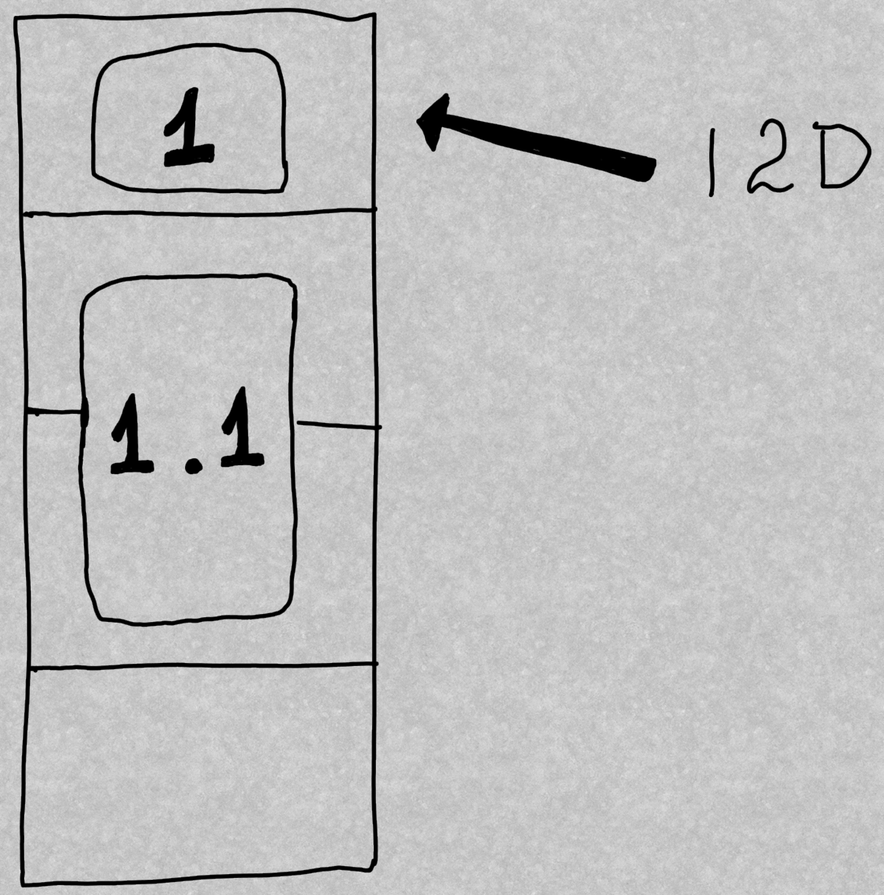

LOCAL VARIABLES
0
String[]
1
VALUE
OPERAND STACK
ILOAD_1
DLOAD_1
ALOAD_1

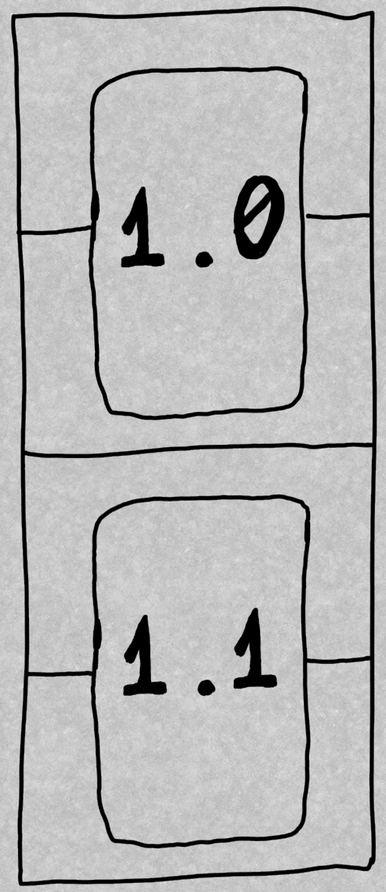

LOCAL VARIABLES
0
String[]
1
VALUE
OPERAND STACK
ILOAD_1
DLOAD_1
ALOAD_1
SOME VALUE

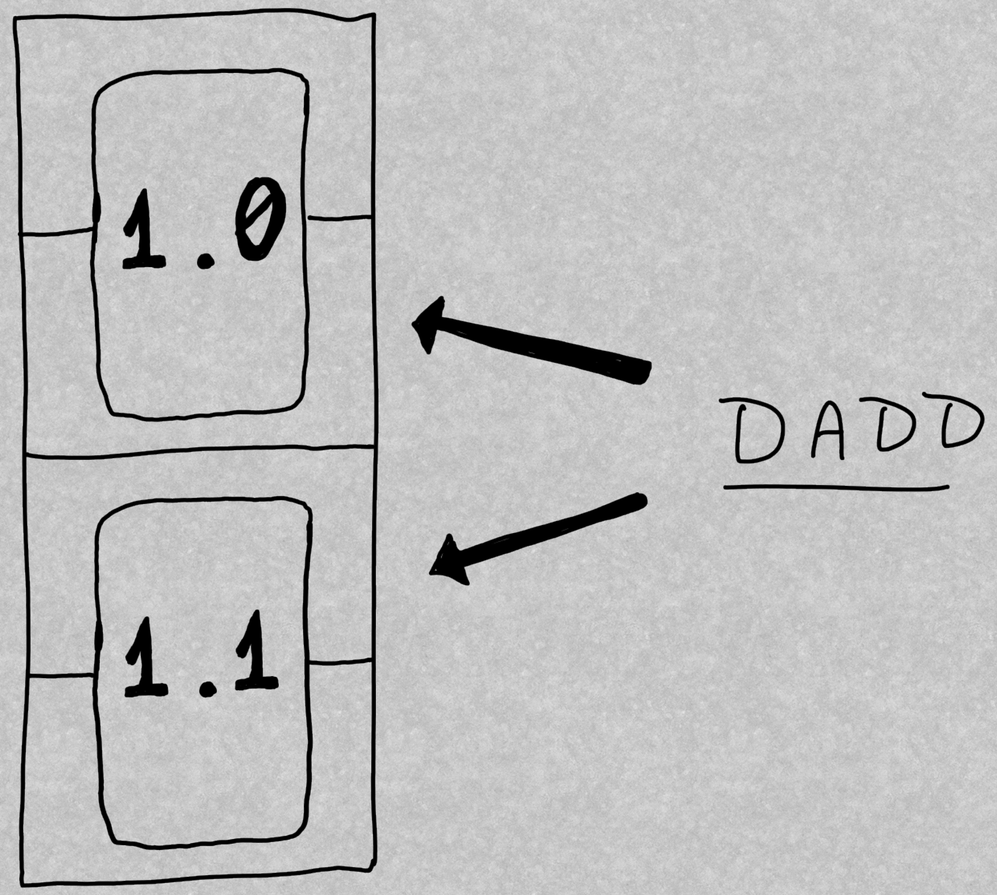

LOCAL VARIABLES
0
String[]
1
VALUE
OPERAND STACK
SOME VALUE

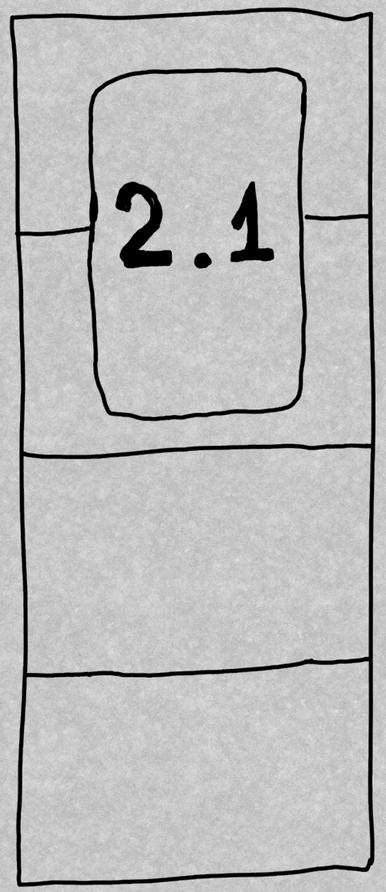

1
1.1
ADD?


1
1.1
IADD ?
DADD ?

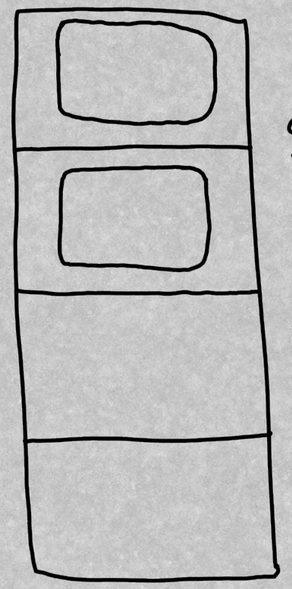

1
1.1
12D


1.0
1.1

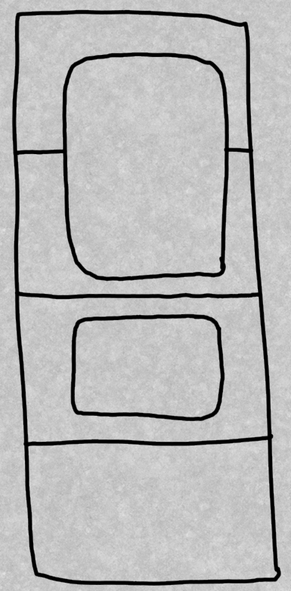

1.0
1.1
DADD


2.1

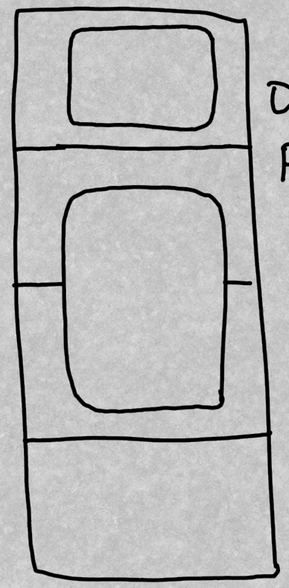
SWAP & DUP



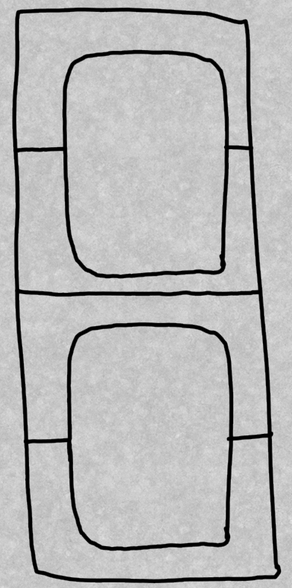
SWAP



DUP2_X1
POP2



DUP_X2
POP



DUP2_X2
POP2

CONDITIONAL BLOCKS

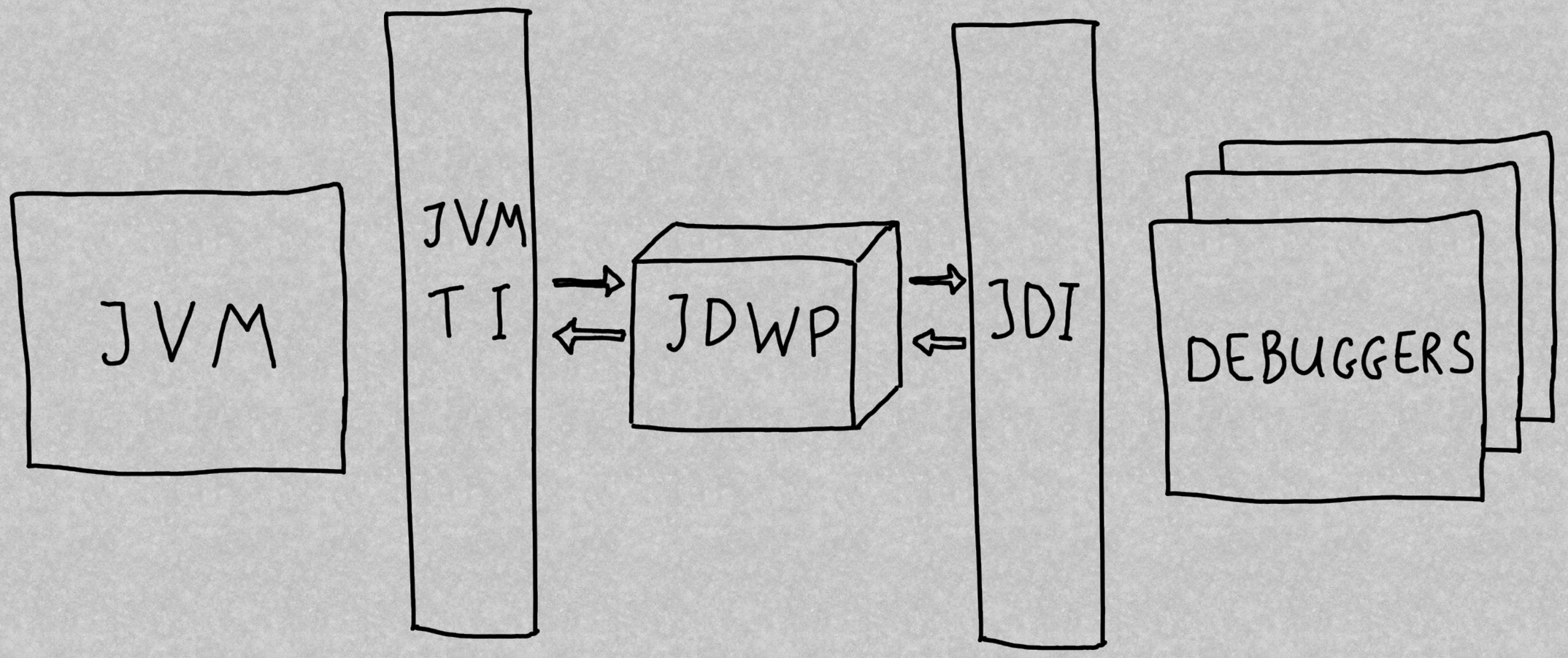
```
...

0: IFGT 6
3: GOTO 10
6: <something>
7: GOTO 11
10: <something else>
11: <continue rest>

...
```

```
Compiled from "Test.java"
public class smthelusive.mova.Test {
  public smthelusive.mova.Test();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."<init>":()V
       4: return

  public static void main(java.lang.String[]);
    Code:
       0: iconst_2
       1: istore_1
       2: getstatic     #7                  // Field java/lang/System.out:Ljava/io/PrintStream;
       5: iload_1
       6: invokevirtual #13                 // Method java/io/PrintStream.println:(I)V
       9: return
}
```

Java type	Type descriptor
`boolean`	`Z`
`char`	`C`
`byte`	`B`
`short`	`S`
`int`	`I`
`float`	`F`
`long`	`J`
`double`	`D`
`Object`	`Ljava/lang/Object;`
`int[]`	`[I`
`Object[][]`	`[[Ljava/lang/Object;`

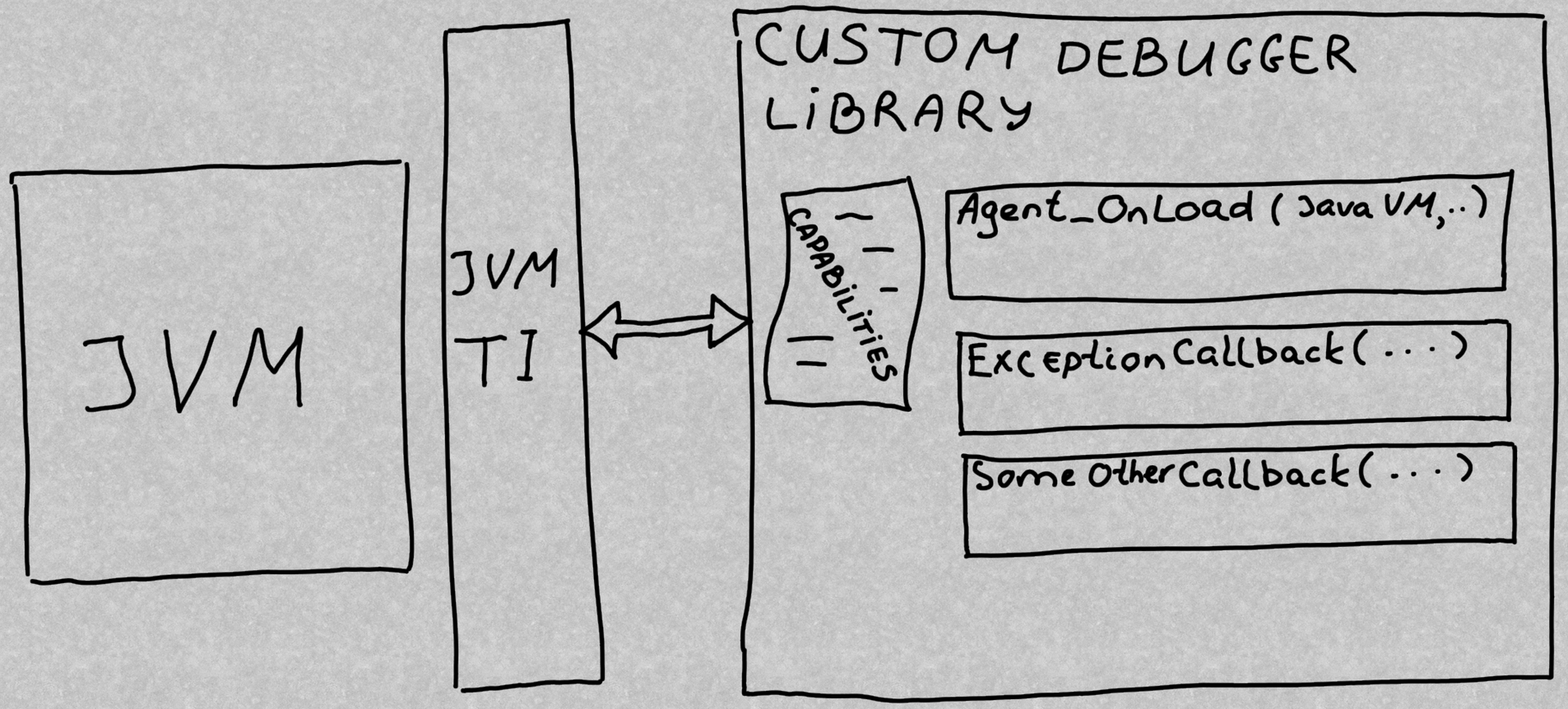
Method declaration in source file	Method descriptor
`void m(int i, float f)`	`(IF)V`
`int m(Object o)`	`(Ljava/lang/Object;)I`
`int[] m(int i, String s)`	`(ILjava/lang/String;)[I`
`Object m(int[] i)`	`([I)Ljava/lang/Object;`

```
Error: Unable to initialize main class WIP
Caused by: java.lang.VerifyError: Bad type on operand stack
Exception Details:
  Location:
    WIP.main([Ljava/lang/String;)V @83: swap
  Reason:
    Type double_2nd (current frame, stack[2]) is not assignable to category1 type
  Current Frame:
    bci: @83
    flags: { }
    locals: { '[Ljava/lang/String;', integer, integer, double, double_2nd, integer }
    stack: { integer, double, double_2nd }
  Bytecode:
    0000000: 1207 1207 603c 1207 1208 1208 6012 0860
    0000010: 1208 6012 0712 0760 6060 1207 1207 6012
    0000020: 0760 1207 6060 1207 1207 1207 6012 0764
    0000030: 6060 1207 601b 683d 1400 0912 0787 6314
    0000040: 000b 6b12 0887 6f4a 1b12 0d60 3605 1207
    0000050: 1400 0e5f 875f 6312 1012 1160 8767 1400
    0000060: 1212 1487 6314 0015 6763 1217 8767 3906
    0000070: 1218 3605 1505 1207 6036 0515 0536 051b
    0000080: 1207 643c 1b36 05b1
```


JVM
JVM
TI
JDWP
JDI
DEBUGGERS

-agentpath

JAVA_TOOL_OPTIONS



JDWP-Handshake

SHODAN

jdwp-handshake

TOTAL RESULTS

134,003

TOP COUNTRIES

China	132,644
Hong ...	1,350
United Sta...	4
Canada	3
Poland	1

More...

View Report View on Map

39.106.236.189 2023-02-14T14:47:42.879241

Aliyun Computing Co., LTD JDWP-Handshake

China, Beijing

39.104.10.88 2023-02-14T14:47:25.581609

Aliyun Computing Co., LTD JDWP-Handshake

China, Beijing

47.104.36.36 2023-02-14T14:46:47.984364

Aliyun Computing Co., LTD JDWP-Handshake

China, Qingdao

3,013,567 code results

Sort: Best match

chrisglass/puppet-hadoop-modules

templates/environment/etc/profile.d/hstack_options.sh

```
3  export HADOOP_NAMENODE_OPTS="-Xdebug -Xrunjdwp
   :transport=dt_socket,address=1044,server=y,suspend=n"
4  export HADOOP_SECONDARYNAMENODE_OPTS="-Xdebug -
   Xrunjdwp:transport=dt_socket,address=1045,server=y,suspend=n"
5  export HADOOP_DATANODE_OPTS="-Xdebug -
   Xrunjdwp:transport=dt_socket,address=1046,server=y,suspend=n"
```

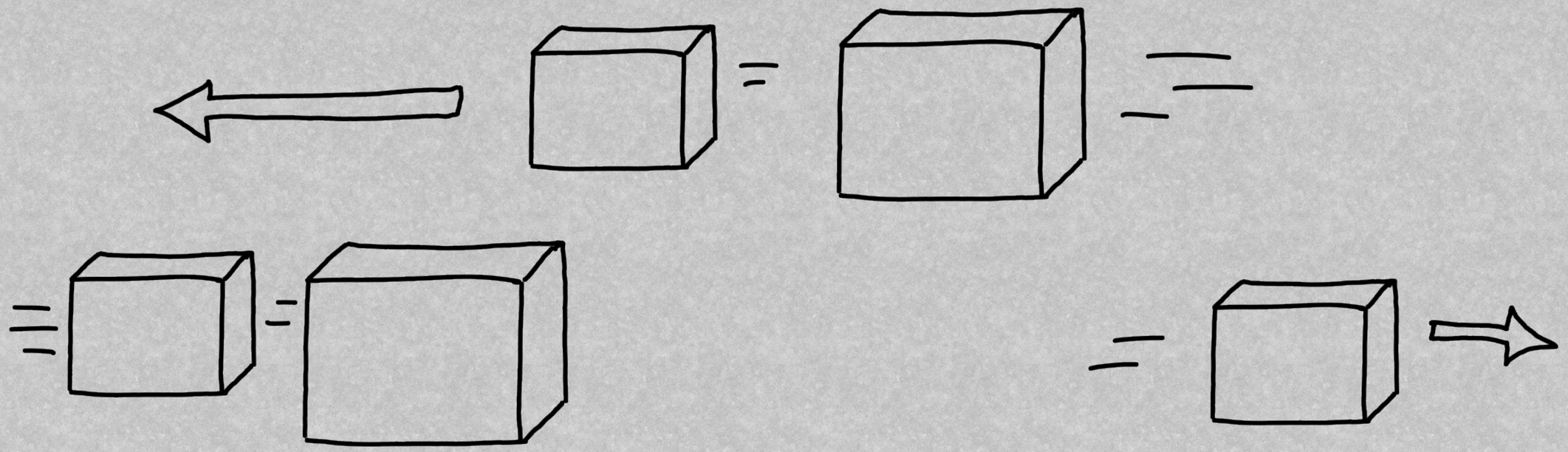
Shell Showing the top two matches Last indexed on Apr 2, 2021

zcharli/distributed-review-sys

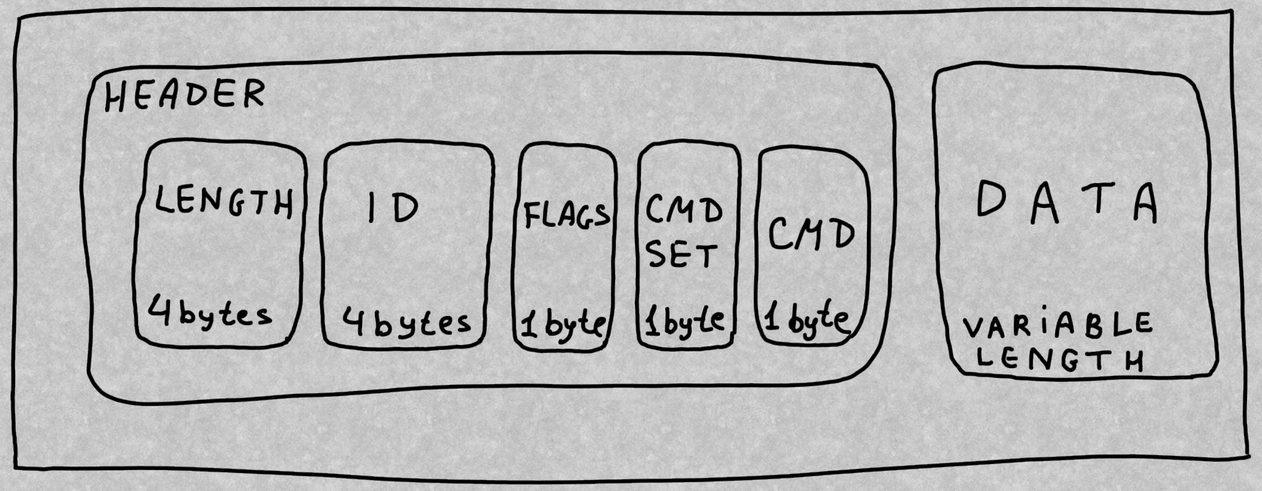
/build.gradle

```
5  apply plugin: 'application'
6
7  applicationDefaultJvmArgs = ["-Xdebug", "-Xrunjdwp
   :transport=dt_socket,server=y,suspend=n,address=8080"]
…
25     classpath = sourceSets.main.runtimeClasspath
26     args '1'
27     jvmArgs '-Xdebug',
28             '-Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8080'
```

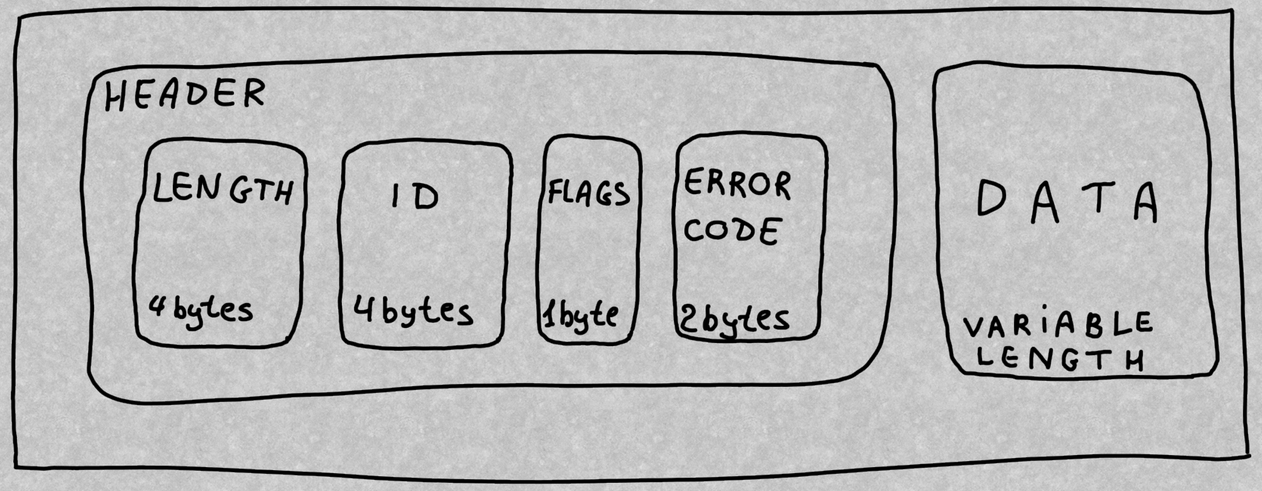
Gradle Showing the top four matches Last indexed on Apr 15, 2021

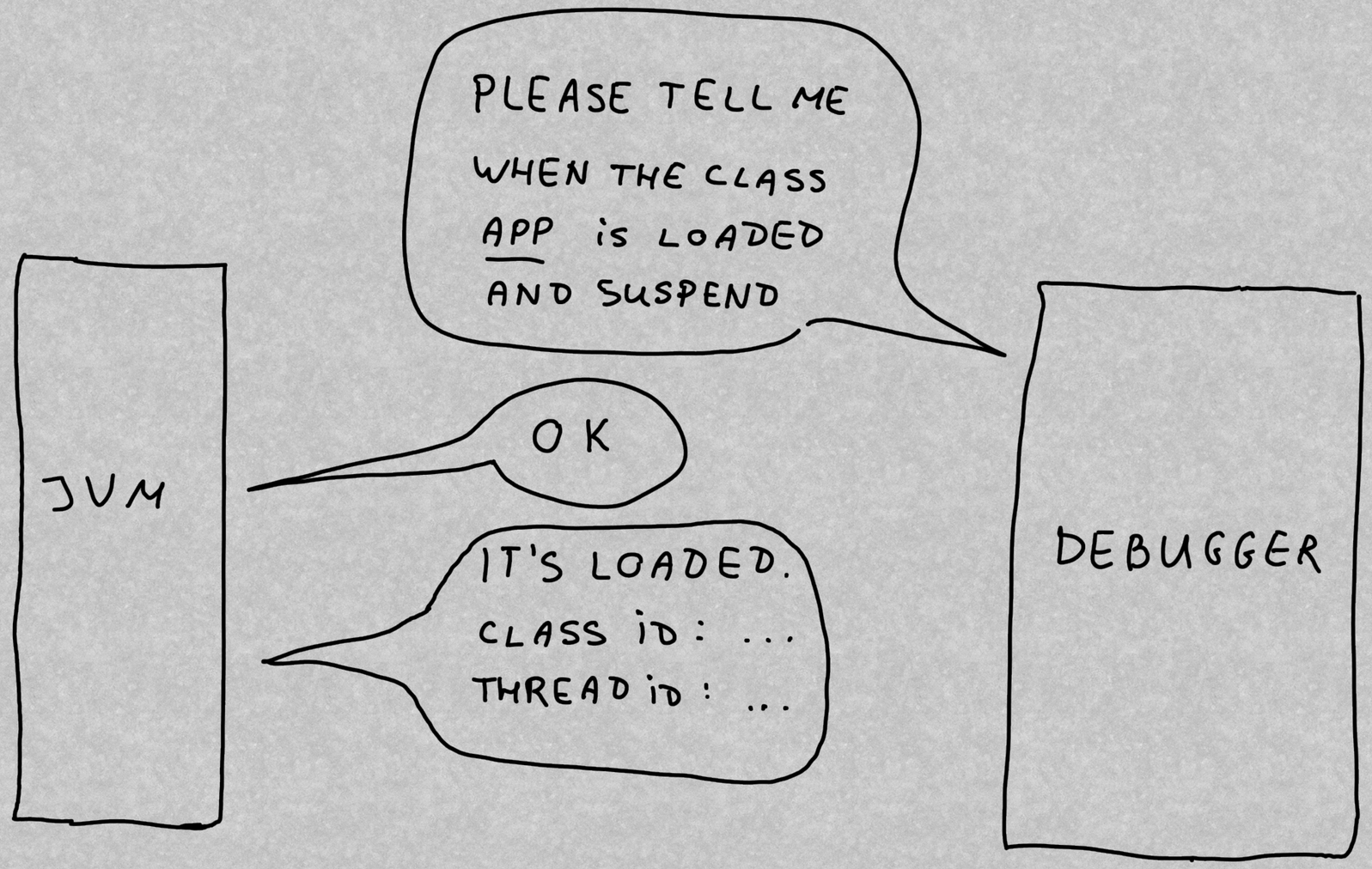


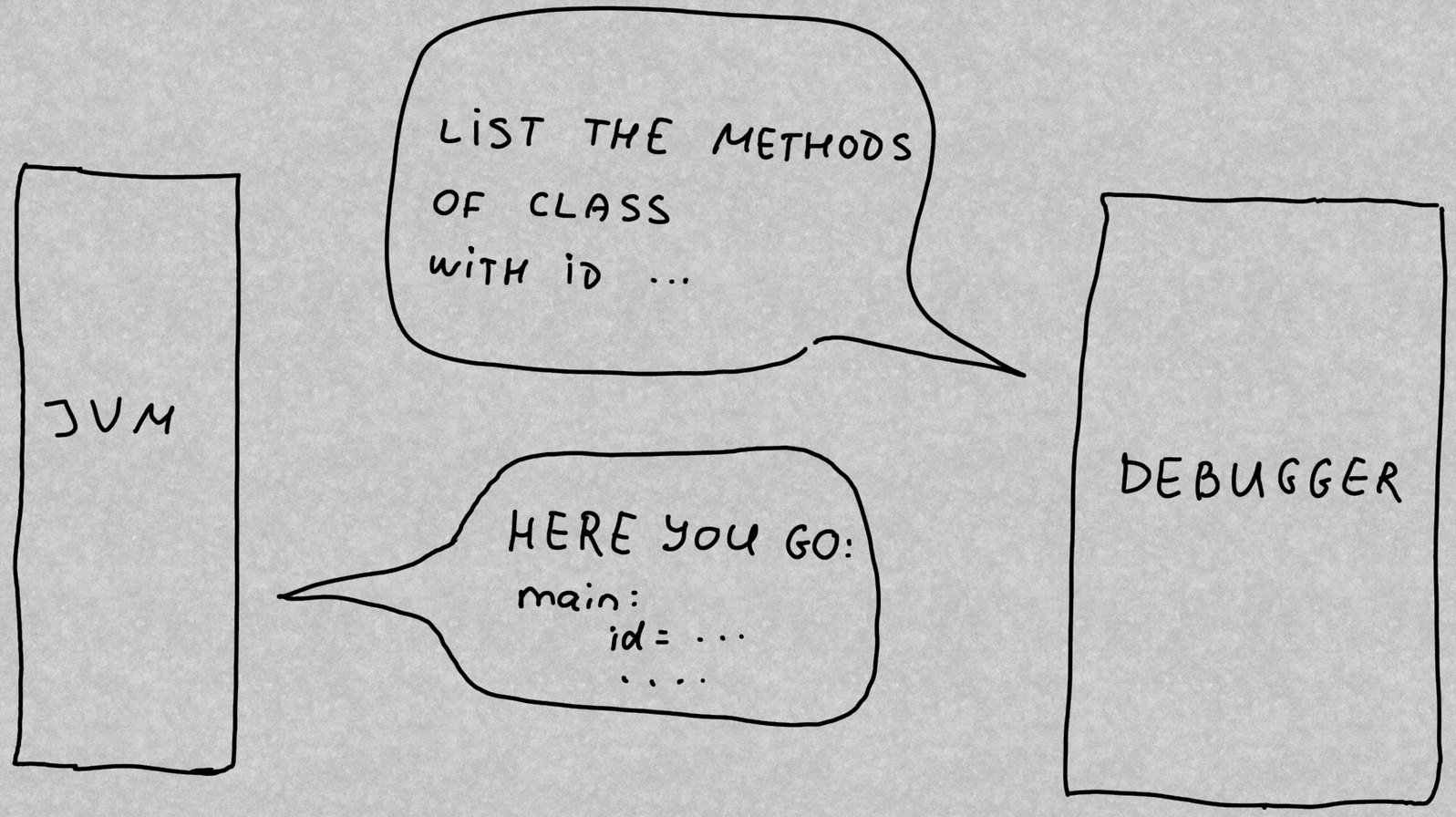
COMMAND PACKET

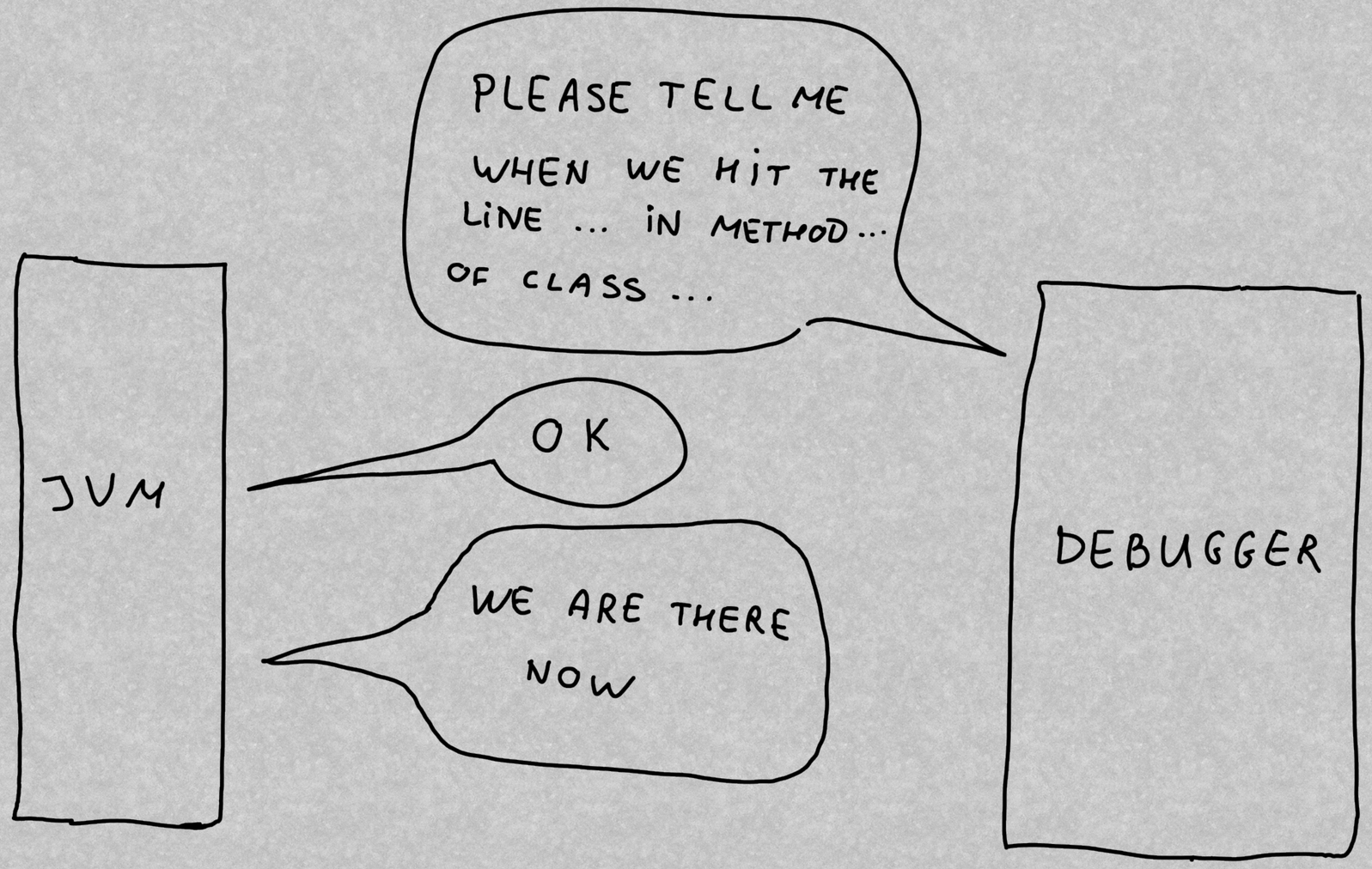


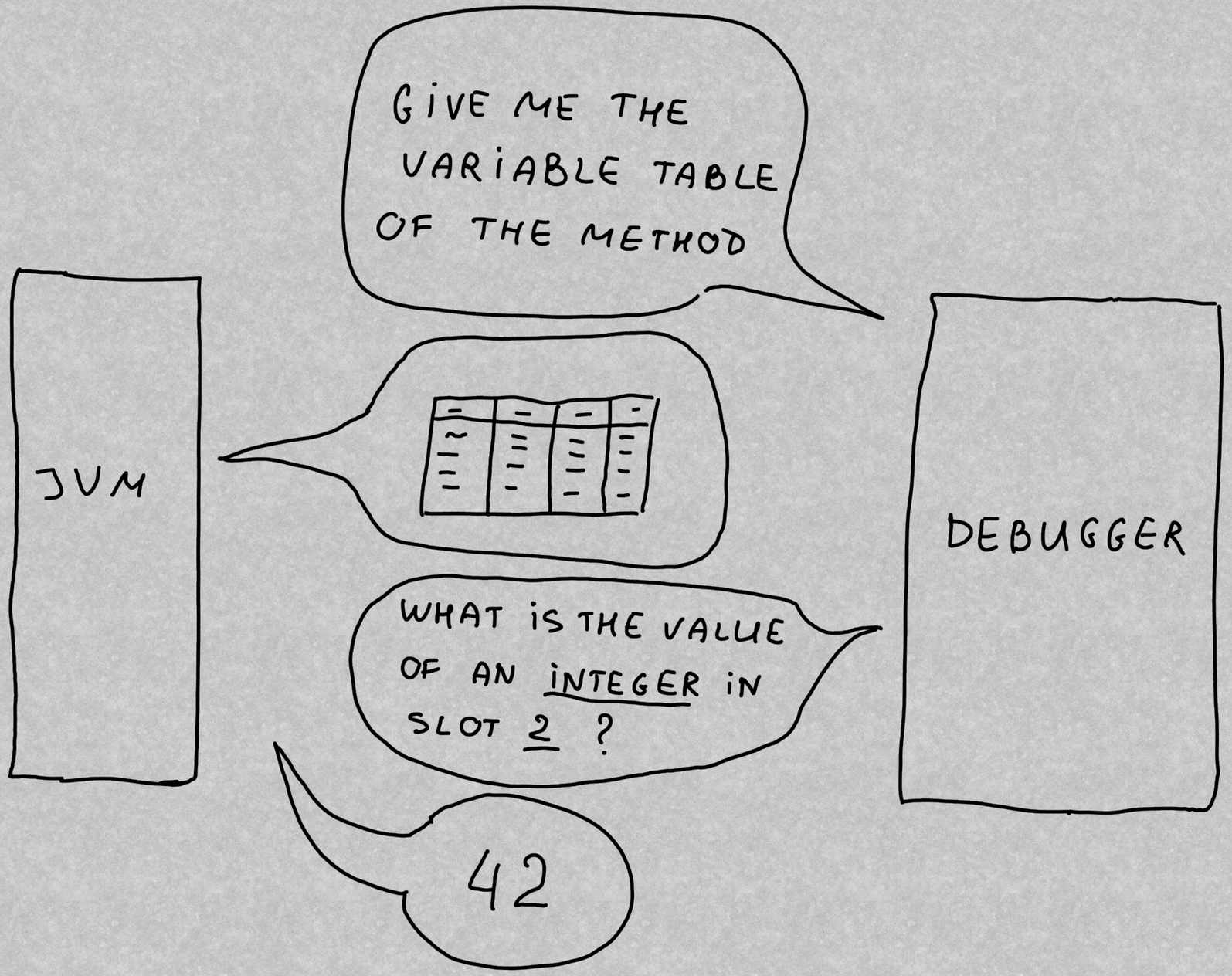
RESPONSE PACKET

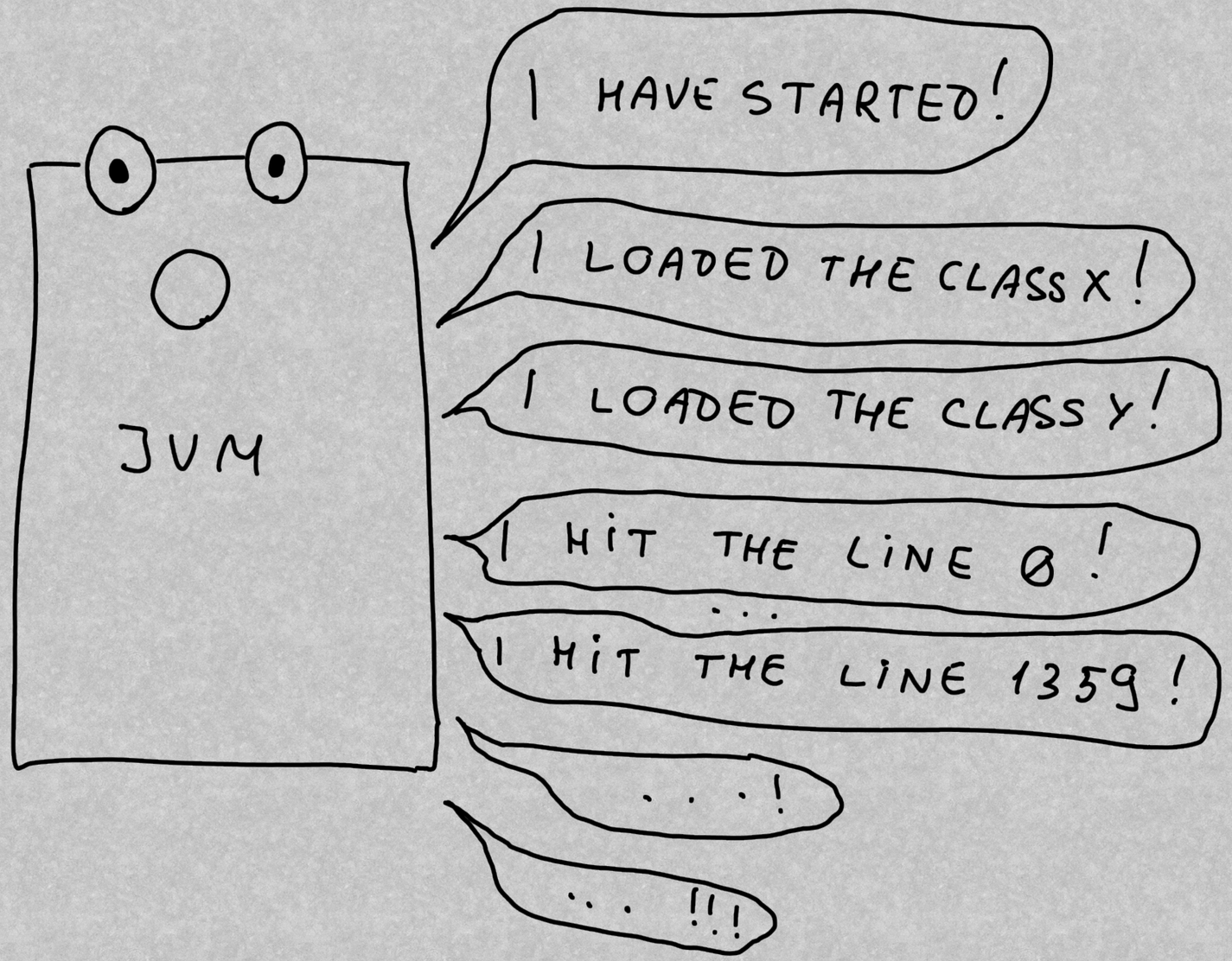


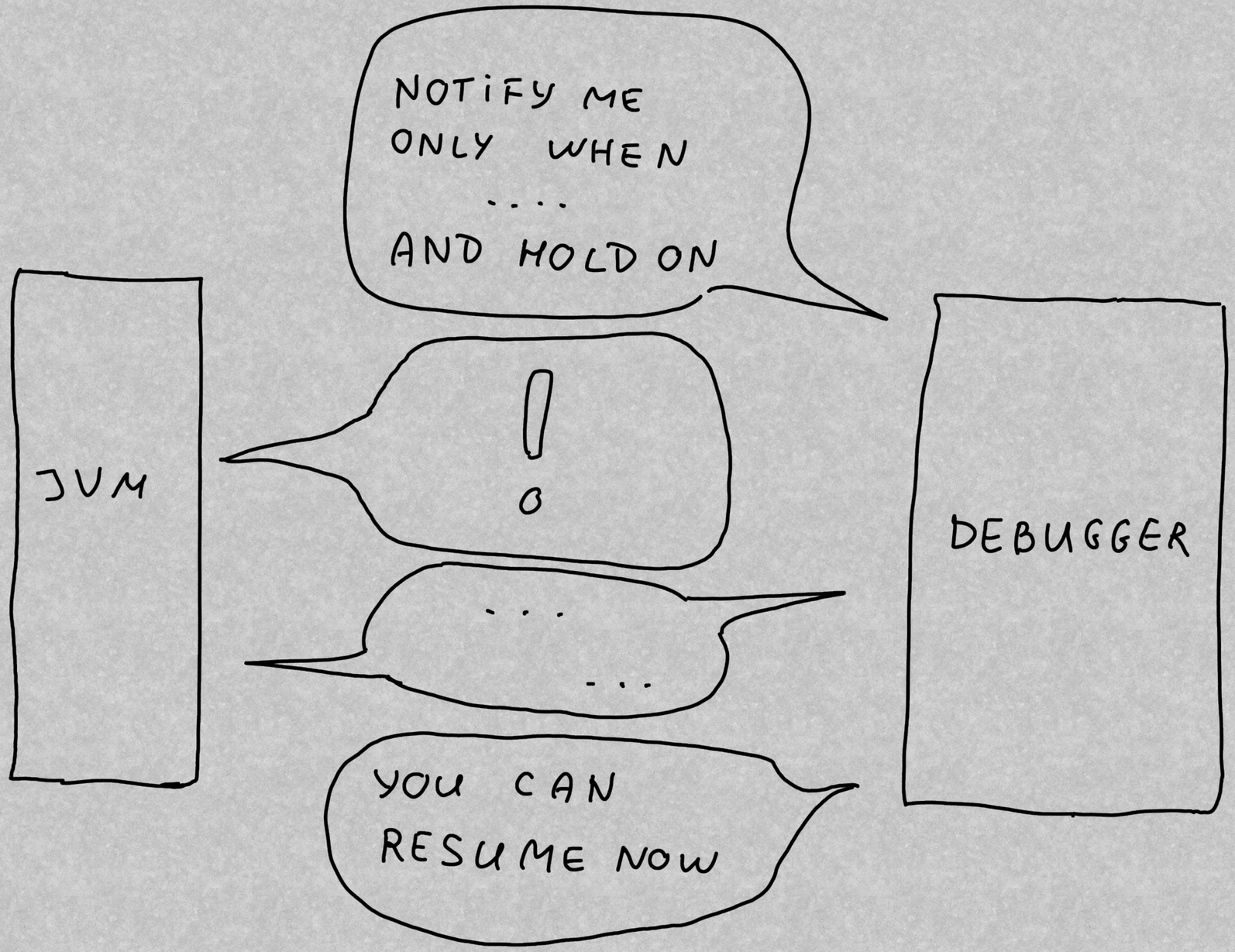

PLEASE TELL ME
WHEN THE CLASS
APP is LOADED
AND SUSPEND
JVM
OK
IT'S LOADED.
CLASS ID: ...
THREAD ID: ...
DEBUGGER

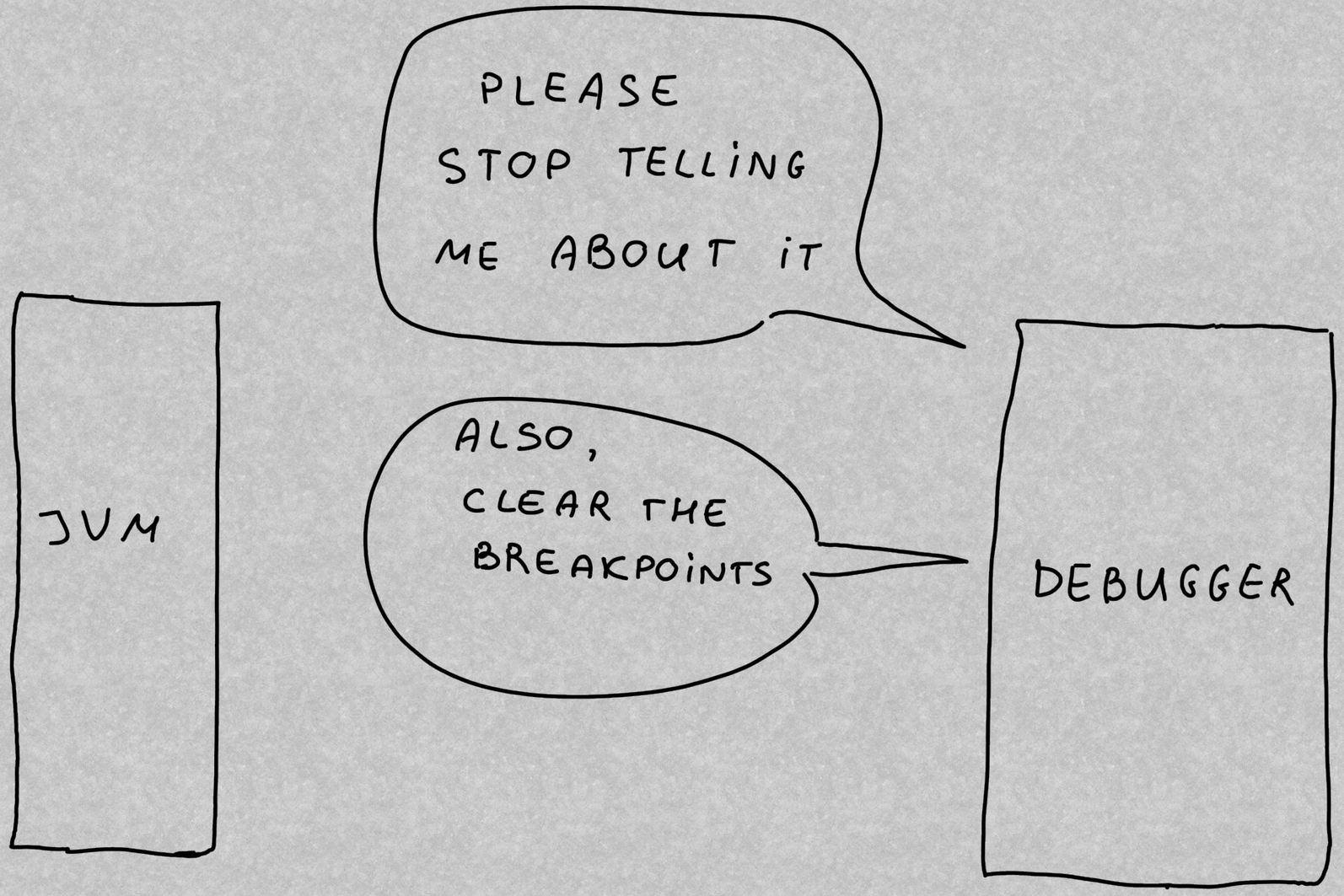

LIST THE METHODS OF CLASS WITH ID ...
JVM
DEBUGGER
HERE YOU GO:
main:
id = ...
....

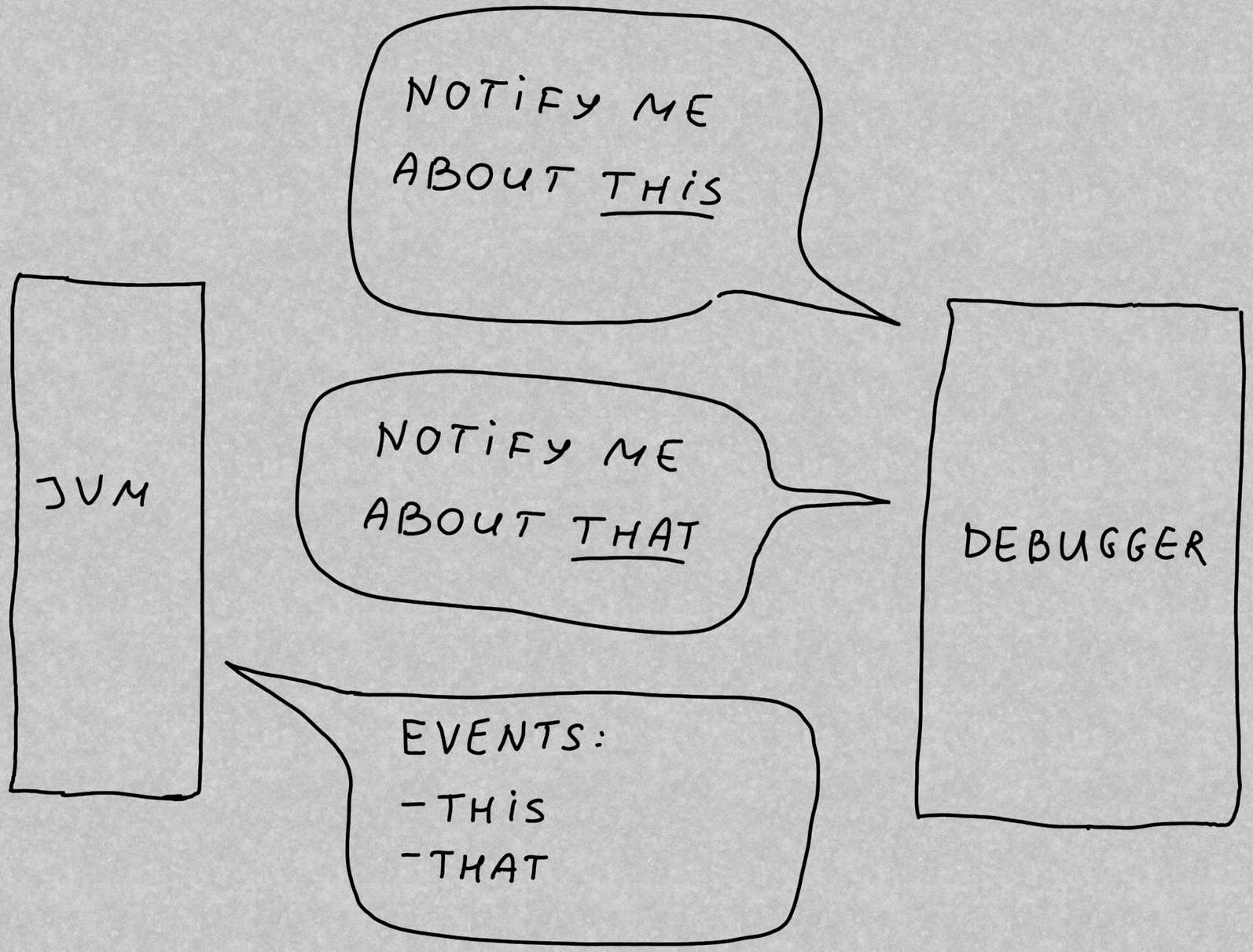

PLEASE TELL ME WHEN WE HIT THE LINE ... IN METHOD ... OF CLASS ...
JVM
OK
WE ARE THERE NOW
DEBUGGER

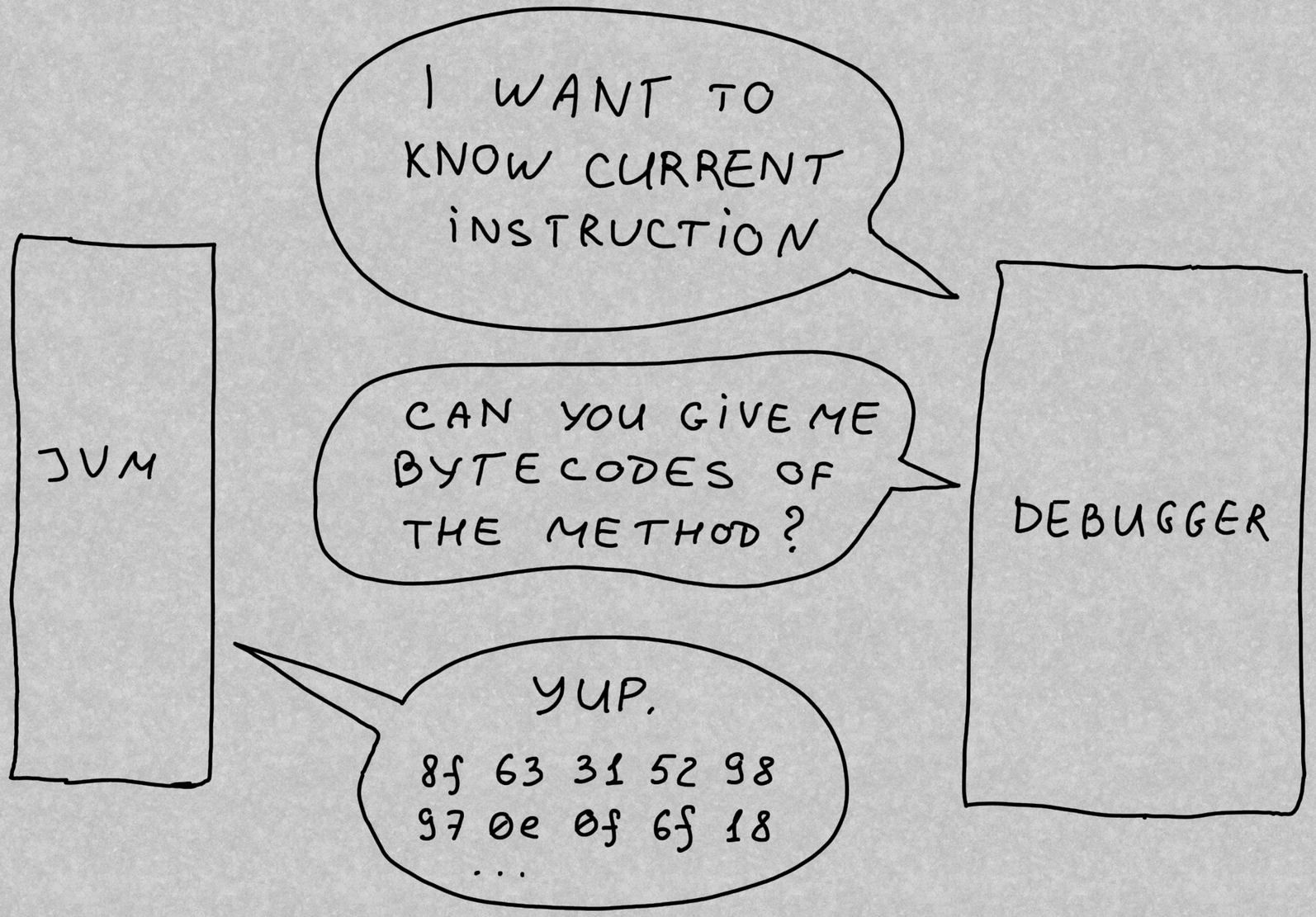

GIVE ME THE VARIABLE TABLE OF THE METHOD
JVM
DEBUGGER
WHAT IS THE VALUE OF AN INTEGER IN SLOT 2 ?
42

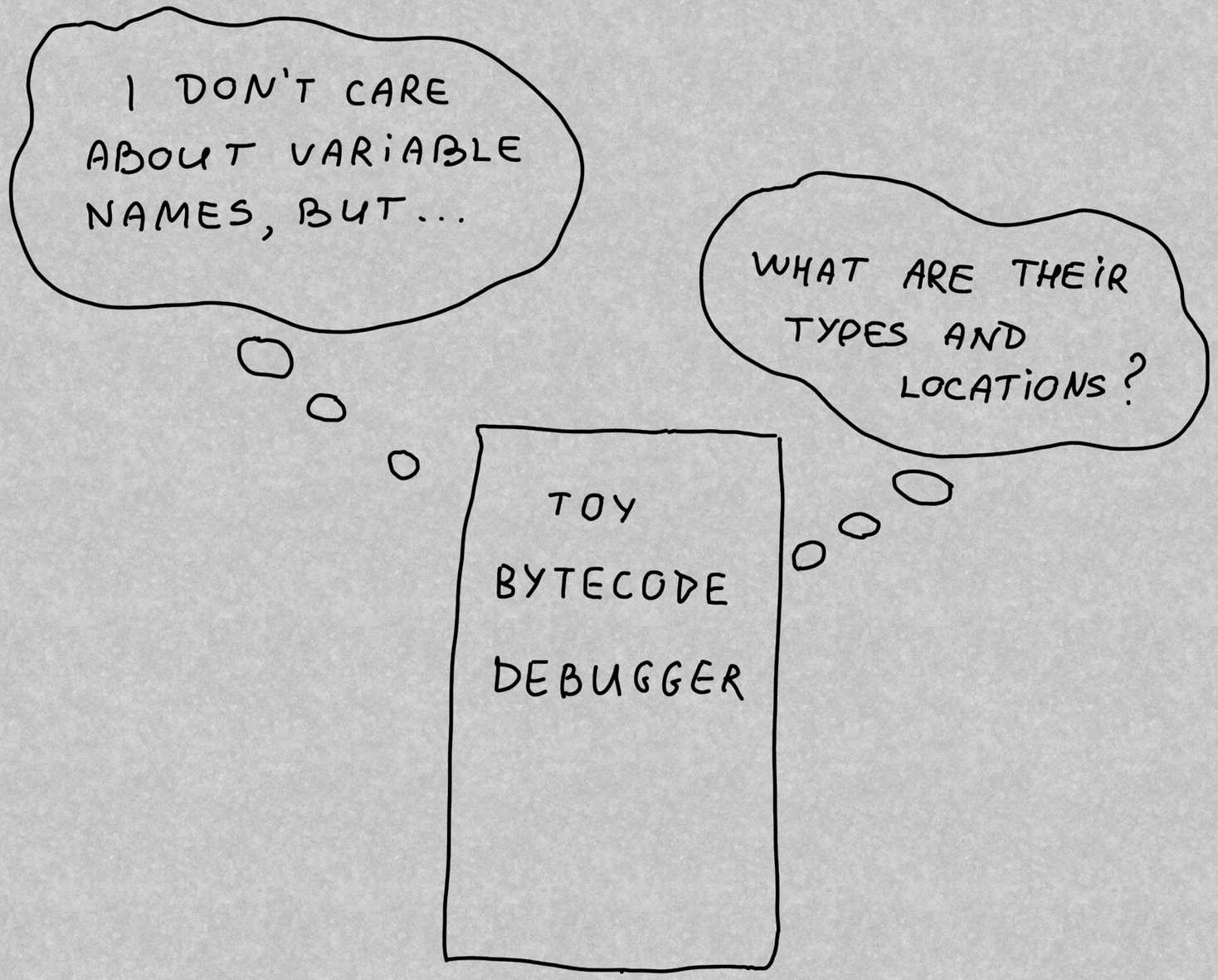

JVM
I HAVE STARTED!
I LOADED THE CLASS X!
I LOADED THE CLASS Y!
I HIT THE LINE 0!
...
I HIT THE LINE 1359!
...!
... !!!


NOTIFY ME ONLY WHEN .... AND HOLD ON
JVM
!
...
...
DEBUGGER
YOU CAN RESUME NOW


PLEASE STOP TELLING ME ABOUT IT
ALSO, CLEAR THE BREAKPOINTS
JVM
DEBUGGER

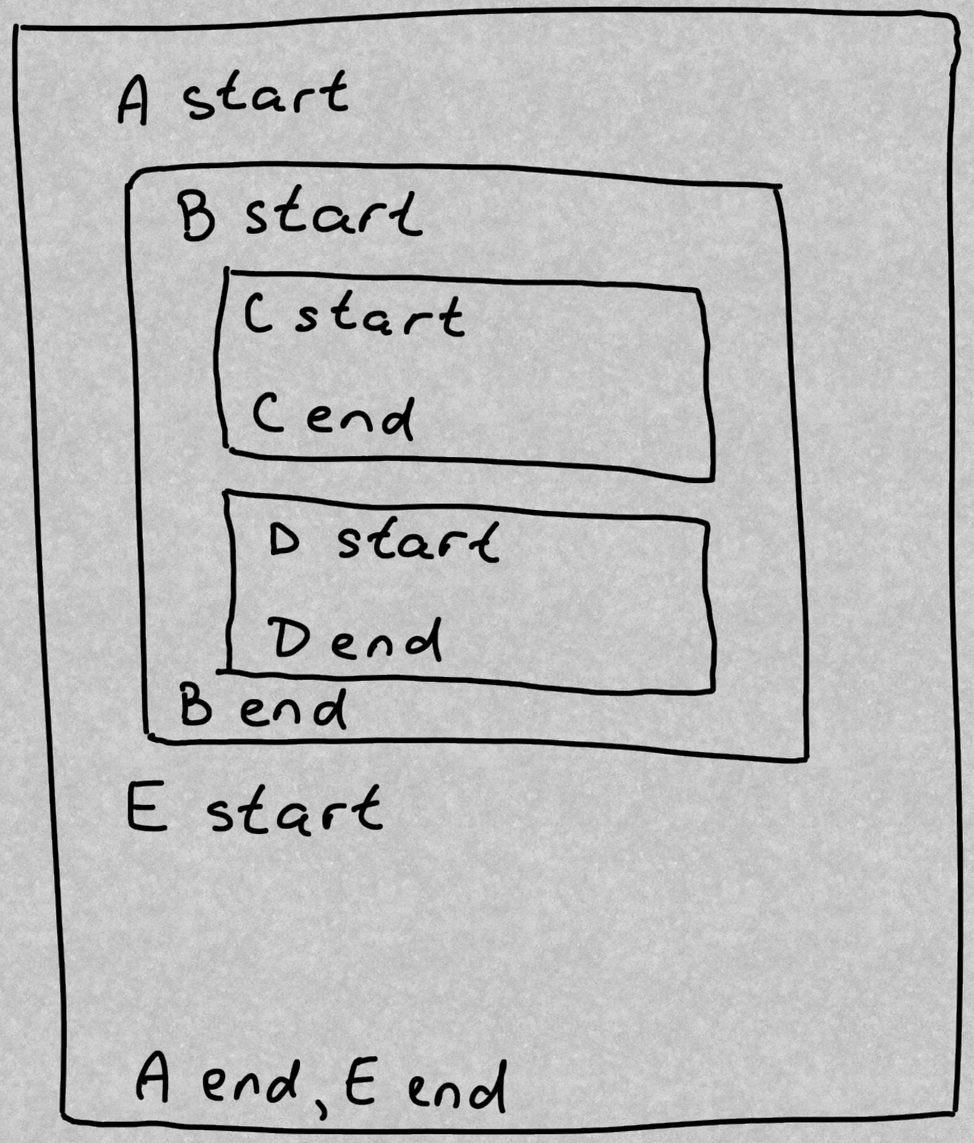

NOTIFY ME ABOUT THIS
JVM
NOTIFY ME ABOUT THAT
DEBUGGER
EVENTS:
-THIS
-THAT

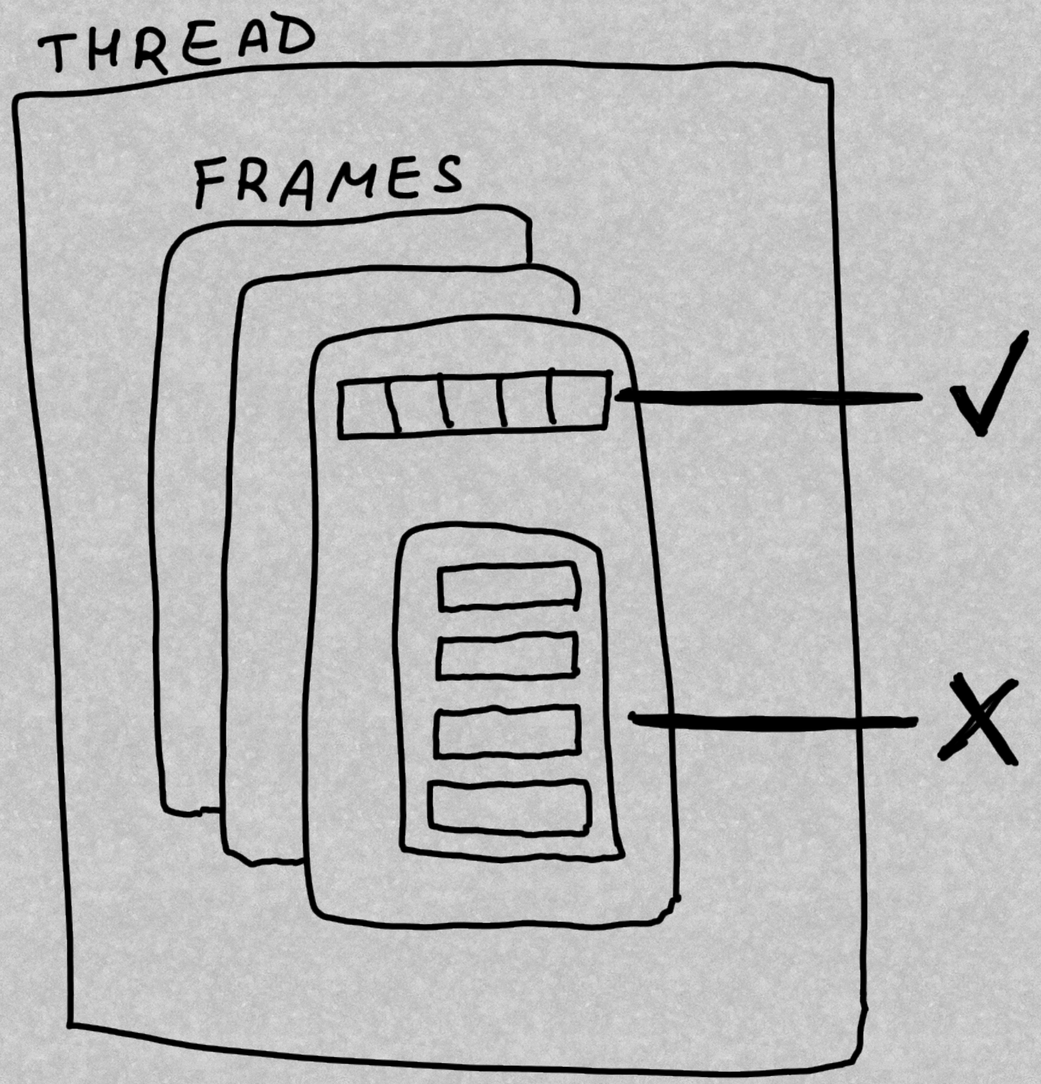

I WANT TO KNOW CURRENT INSTRUCTION
JVM
CAN YOU GIVE ME BYTECODES OF THE METHOD?
DEBUGGER
YUP.
8f 63 31 52 98
97 0e 0f 6f 18
...

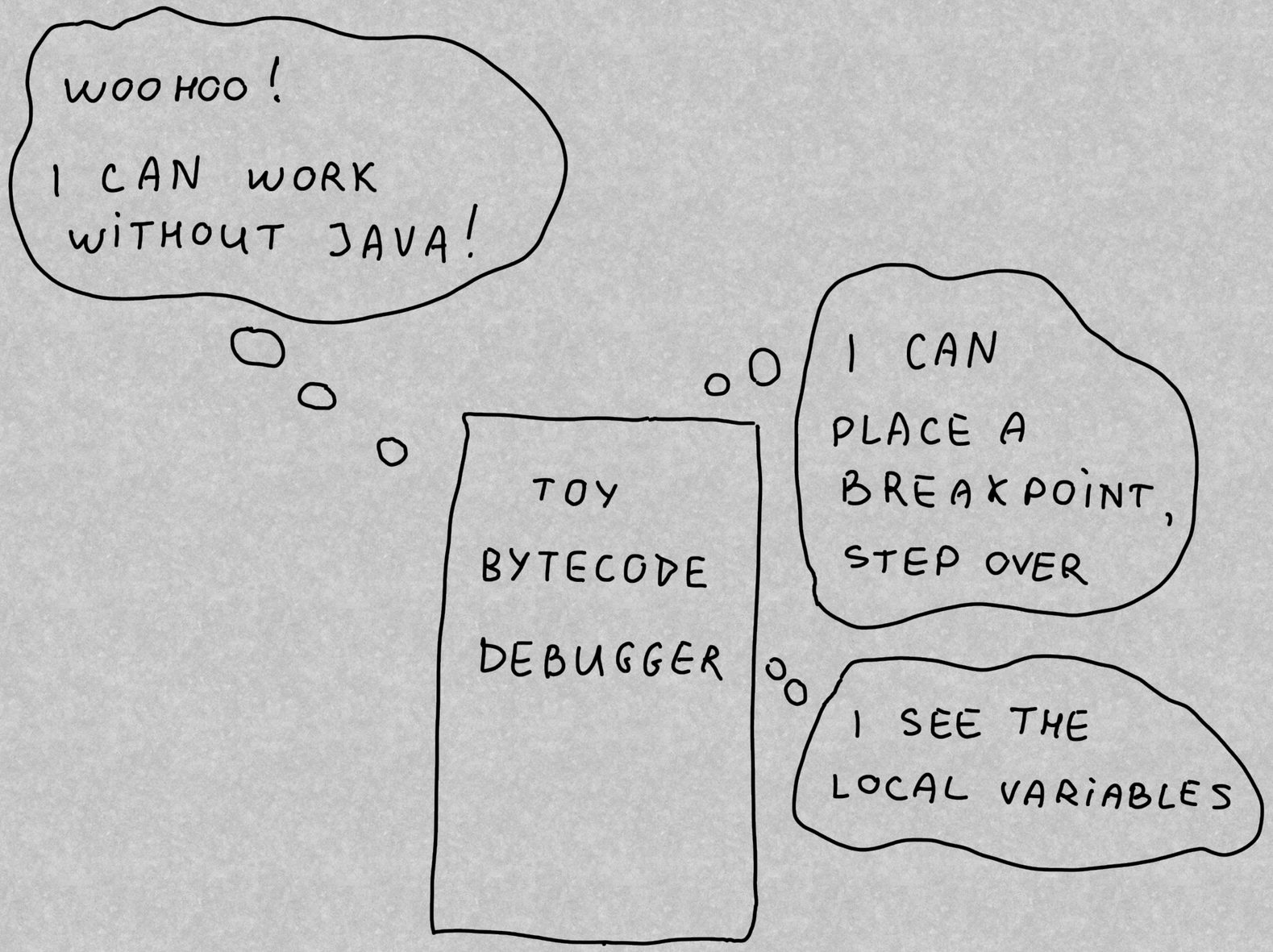

I DON'T CARE ABOUT VARIABLE NAMES, BUT...
WHAT ARE THEIR TYPES AND LOCATIONS?
TOY BYTECODE DEBUGGER

```
Compiled from "TheDebuggee.java"
```

```
    LineNumberTable:
      line 5: 0
      line 6: 2
      line 7: 5
      line 8: 12
      line 9: 15
      line 10: 19
      line 11: 26
      line 12: 38
    LocalVariableTable:
      Start  Length  Slot  Name   Signature
          0      39     0  args   [Ljava/lang/String;
          2      37     1     a   I
         26      13     2  test   Ljava/lang/String;
```


A start
B start
C start
C end
D start
D end
B end
E start
A end, E end


THREAD
FRAMES


WOOHOO!
I CAN WORK WITHOUT JAVA!
TOY BYTECODE DEBUGGER
I CAN PLACE A BREAKPOINT, STEP OVER
I SEE THE LOCAL VARIABLES

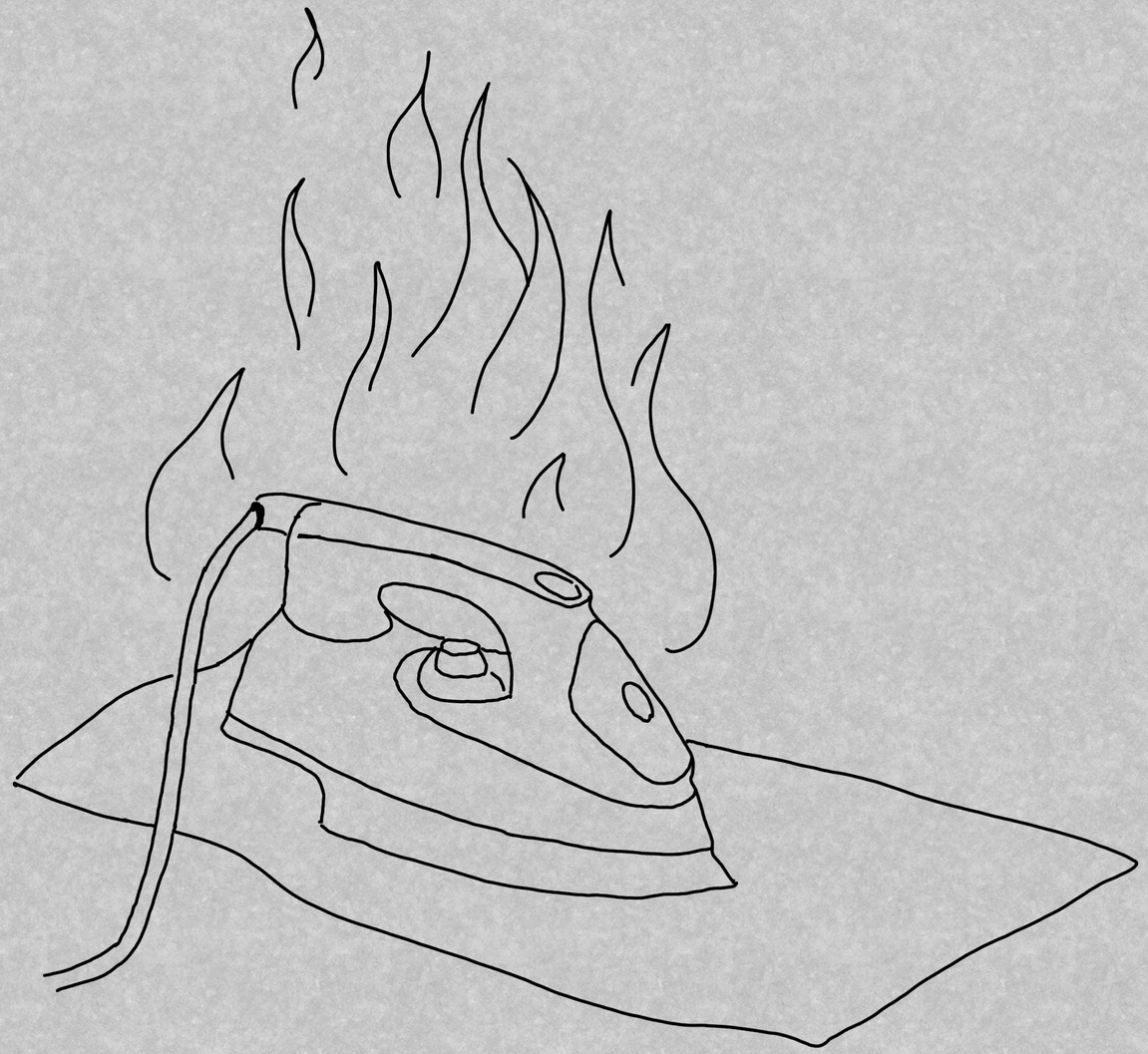
DEBUGGING IS SLOW

- compilation with debug mode on takes more effort
- execution engine can be forced to interpret every single instruction
- debugger has to suspend JVM often to access some info
- depending on the debugger's design there might be an event overhead

JDB

```
nataliiadziubenko@Xebia-Mac src % jdb -attach 127.0.0.1:8000
Set uncaught java.lang.Throwable
Set deferred uncaught java.lang.Throwable
Initializing jdb ...

VM Started: > No frames on the current call stack

main[1] stop at smthelusive.debyter.TestJDB:10
Deferring breakpoint smthelusive.debyter.TestJDB:10.
It will be set after the class is loaded.
main[1] run
> Set deferred breakpoint smthelusive.debyter.TestJDB:10

Breakpoint hit: "thread=main", smthelusive.debyter.TestJDB.main(), line=10 bci=8
10                  a += 1;

main[1] locals
Method arguments:
args = instance of java.lang.String[0] (id=443)
Local variables:
a = 2
b = "string"
main[1]
```

DON’T LEAVE REMOTE DEBUGGING ENABLED



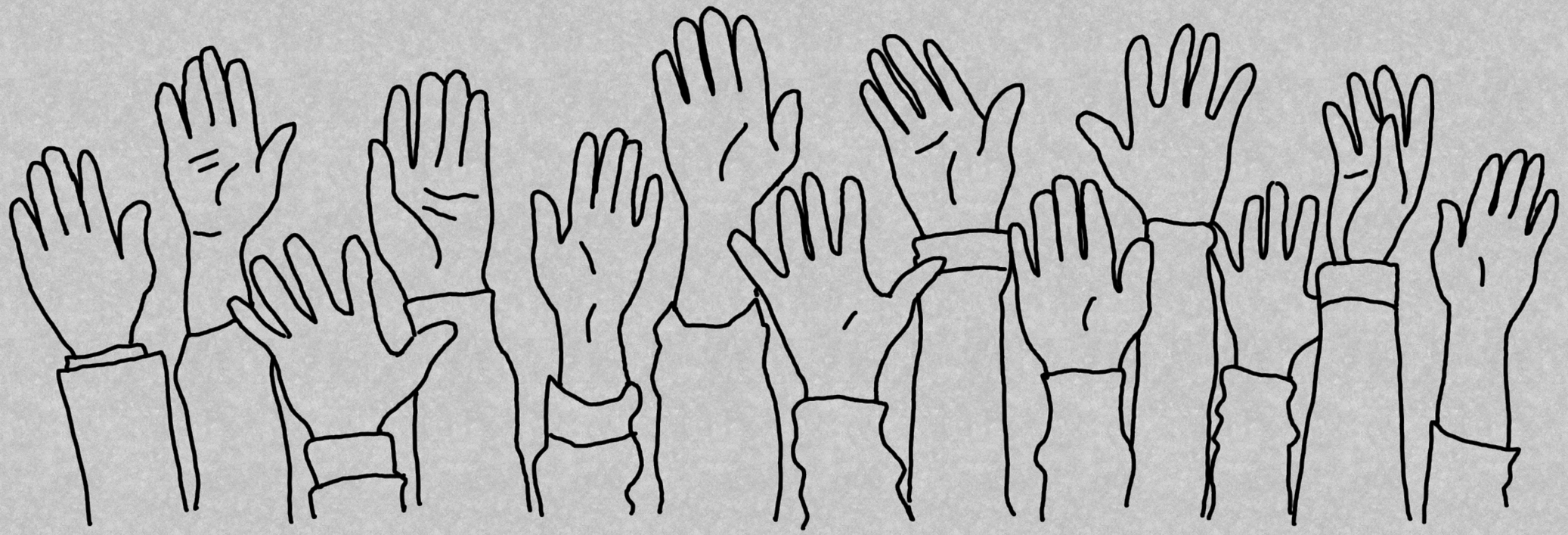
THROUGH JDWP ANYONE CAN:

- set values
- instantiate new objects
- invoke methods
- see the values in runtime

...

- redefine classes!

LET'S TALK!



@worth_exploring