State-of-the-Art Debugging with Jidebug

István Forgács and András Milassin

Outline

Debugging is difficult

Traditional debugging

Reverse debugging

Passive vs Active debugging

Comparison debugging

Influence debugging

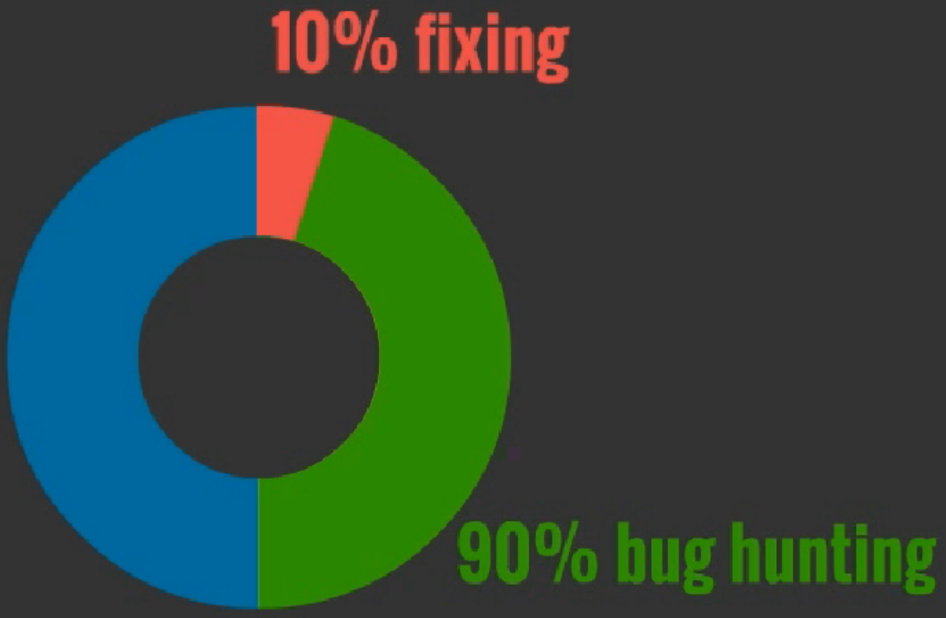
Process of debugging

Feature location

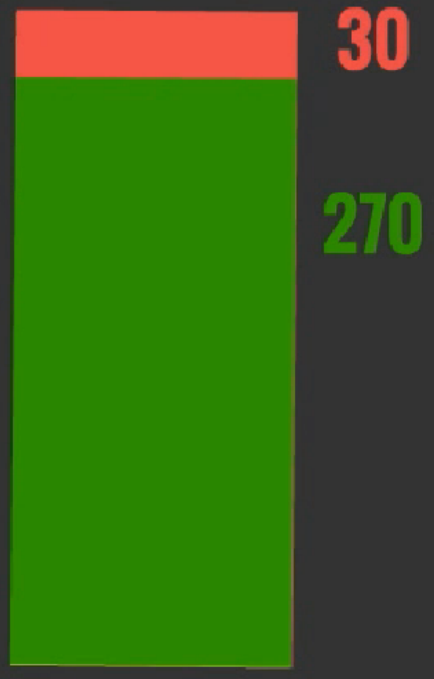
jidebug

ACTIVE DEBUGGER

Introduction



debugging cost



billion USD

Goal of debugging

to find the bug

but how debugging methods help?

debugging methods can be **active** and **passive**

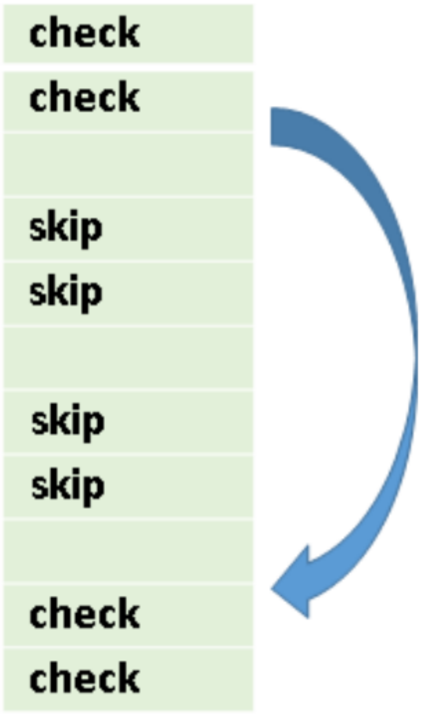
Active and passive debugging

active debuggers reduce the code size to be investigated

passive debuggers only show data about a given execution step

active debugging

```
while (numbers[i] < pivot) {
    i++;
}
while (numbers[j] > pivot) {
    j--;
}
if (i <= j) {
    exchange(numbers, i, j);
}
i++;
j--;
```



check

check

skip

skip

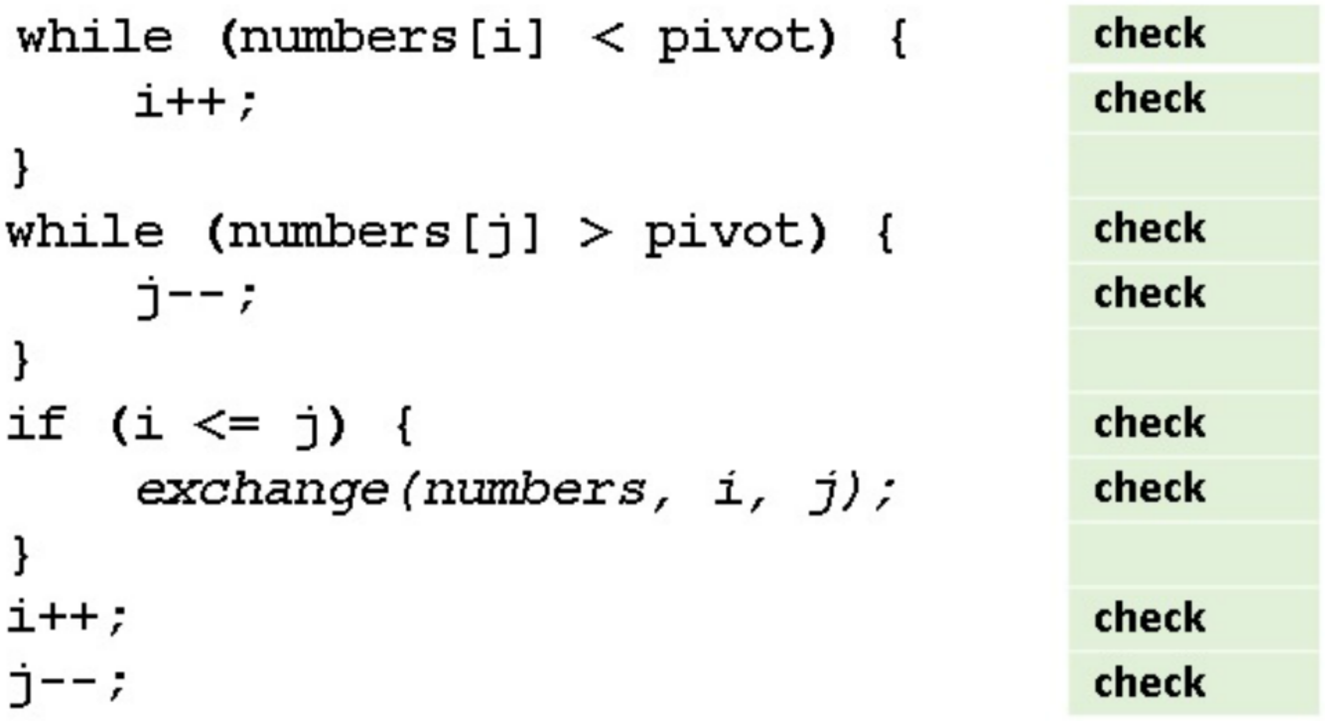
skip

skip

check

check

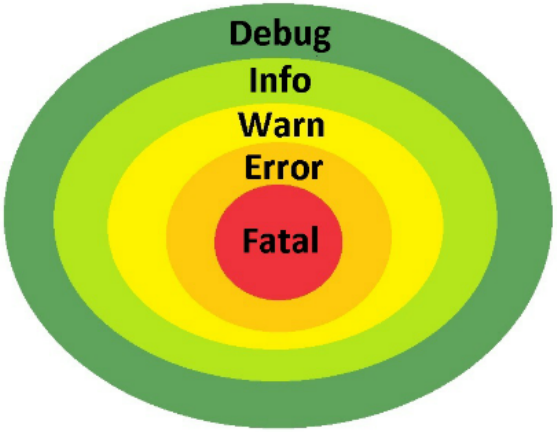
passive debugging



```
while (numbers[i] < pivot) {              check
    i++;                                  check
}
while (numbers[j] > pivot) {              check
    j--;                                  check
}
if (i <= j) {                             check
    exchange(numbers, i, j);              check
}
i++;                                      check
j--;                                      check
```

Logging

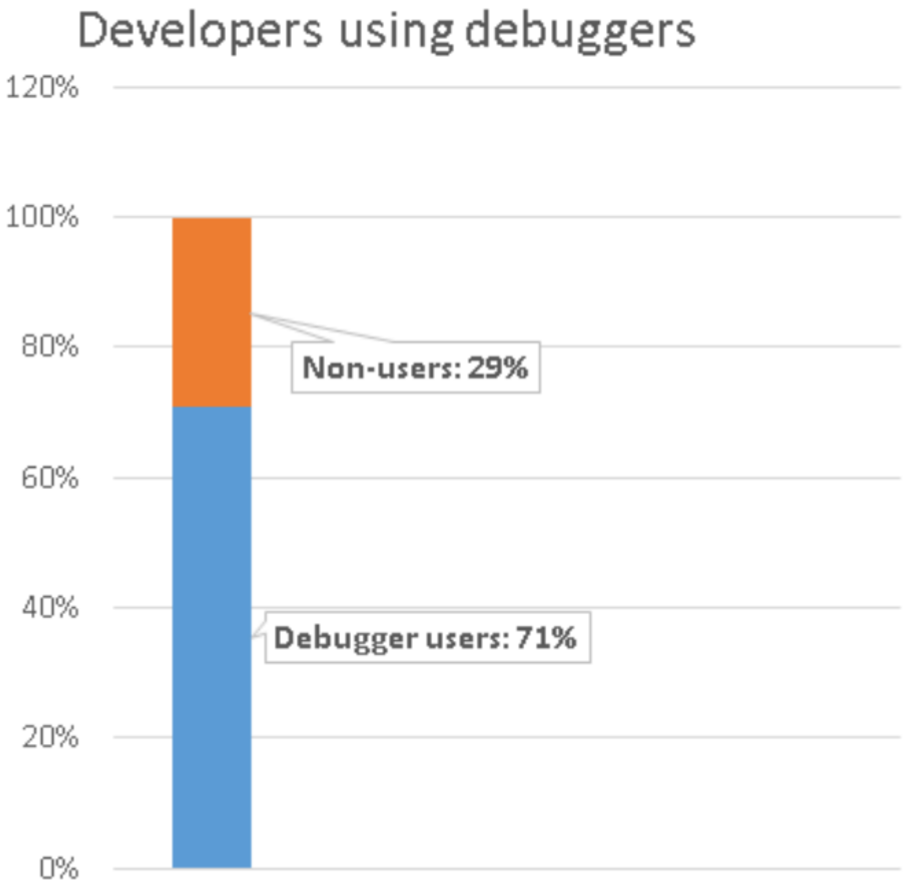
System.out.print



advantages

disadvantages

Traditional debugging



advantages

disadvantages

Reverse debugging

you can start debugging backward from the failure

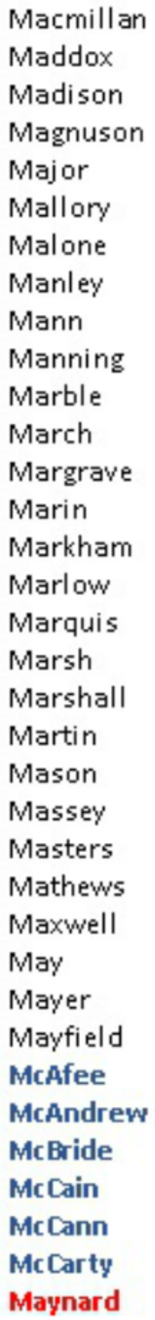
hester
18
22
21
102
23

Reverse debugging

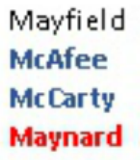
why developers don’t use

Reverse debugging

Active Debugging - input reduction



Macmillan
Maddox
Madison
Magnuson
Major
Mallory
Malone
Manley
Mann
Manning
Marble
March
Margrave
Marin
Markham
Marlow
Marquis
Marsh
Marshall
Martin
Mason
Massey
Masters
Mathews
Maxwell
May
Mayer
Mayfield
McAfee
McAndrew
McBride
McCain
McCann
McCarty
Maynard



Mayfield
McAfee
McCarty
Maynard

Active debugging - hypothesis

Malone	✓
Mayfield	✓
McAfee	✗
McCarty	✗
Maynard	✓

passive and active debugging together

Comparison debugging

we introduced and implemented in Jidebug

we discovered it can be used for debugging and code understanding

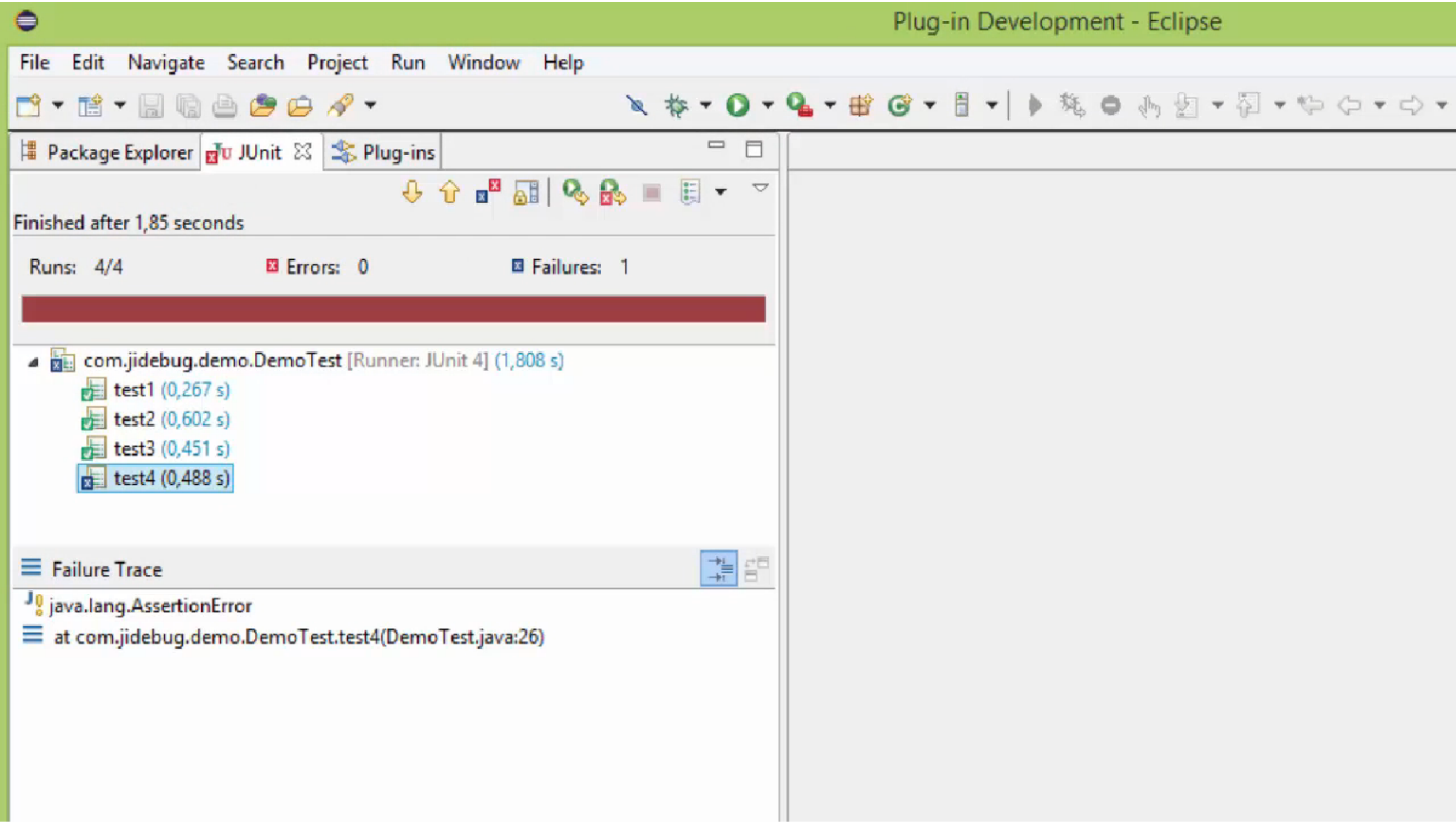
Comparison debugging

✗

✓

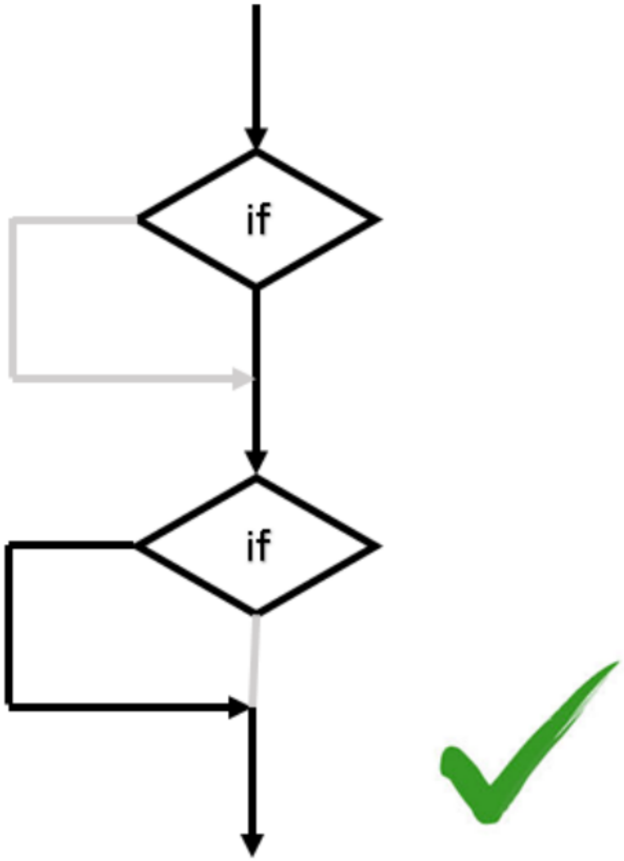
there is solution based on execution comparison

standard



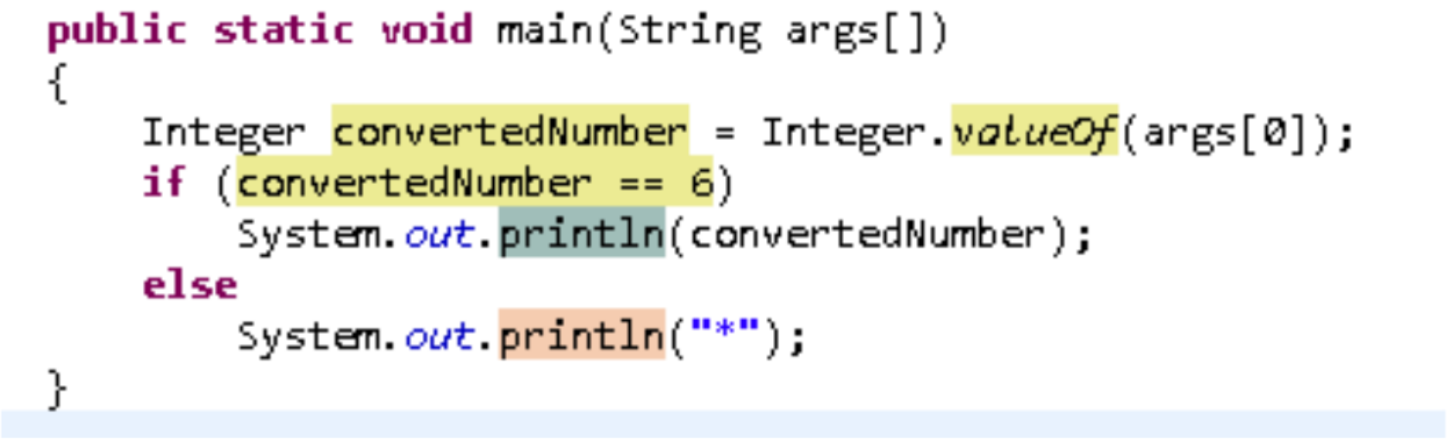
```
Finished after 1,85 seconds

Runs:  4/4          Errors:  0          Failures:  1

com.jidebug.demo.DemoTest [Runner: JUnit 4] (1,808 s)
    test1 (0,267 s)
    test2 (0,602 s)
    test3 (0,451 s)
    test4 (0,488 s)

java.lang.AssertionError
at com.jidebug.demo.DemoTest.test4(DemoTest.java:26)
```

execution trace as standard



compare this and the failed one

comparison result



```
public static void main(String args[])
{
    Integer convertedNumber = Integer.valueOf(args[0]);
    if (convertedNumber == 6)
        System.out.println(convertedNumber);
    else
        System.out.println("*");
}
```

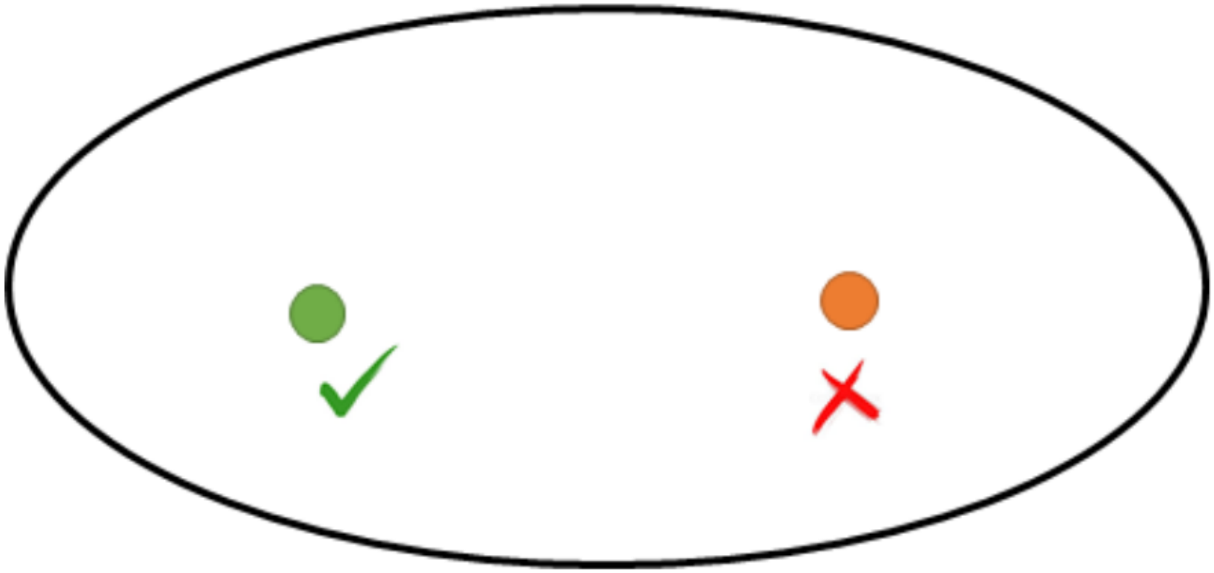
bug hunting

we should start from the differences

consider the first difference

equivalence partition

each input would require similar execution trace



Example

javaone.dat

javatwo.dat

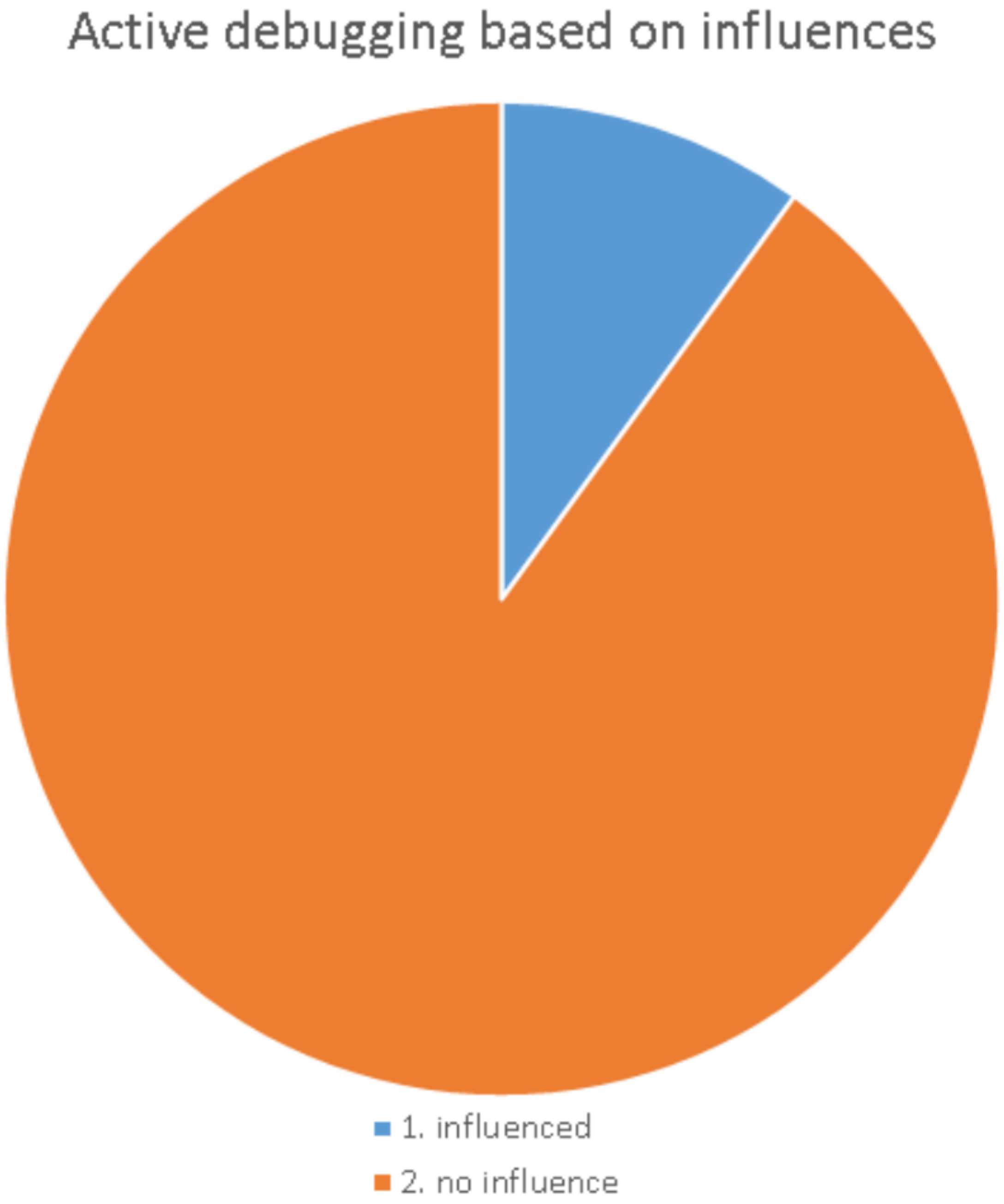
Comparison debugging

advantages - disadvantages

example

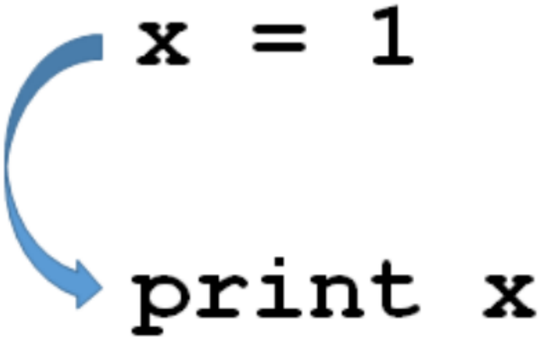
Hill, Adam	$8100
Green-Scott, Gabriel	$7950
McConnell, May	$7200
Perry, Kelly	$8800
Scott, Sally	$8500

Influence debugging



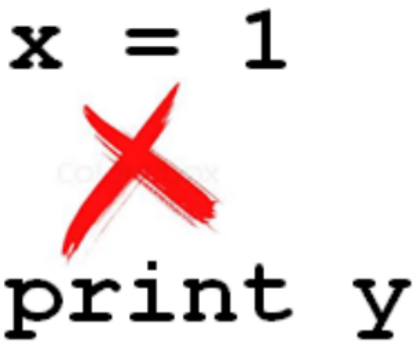
What is an influence?

Influence:



```
x = 1

print x
```

No influence:



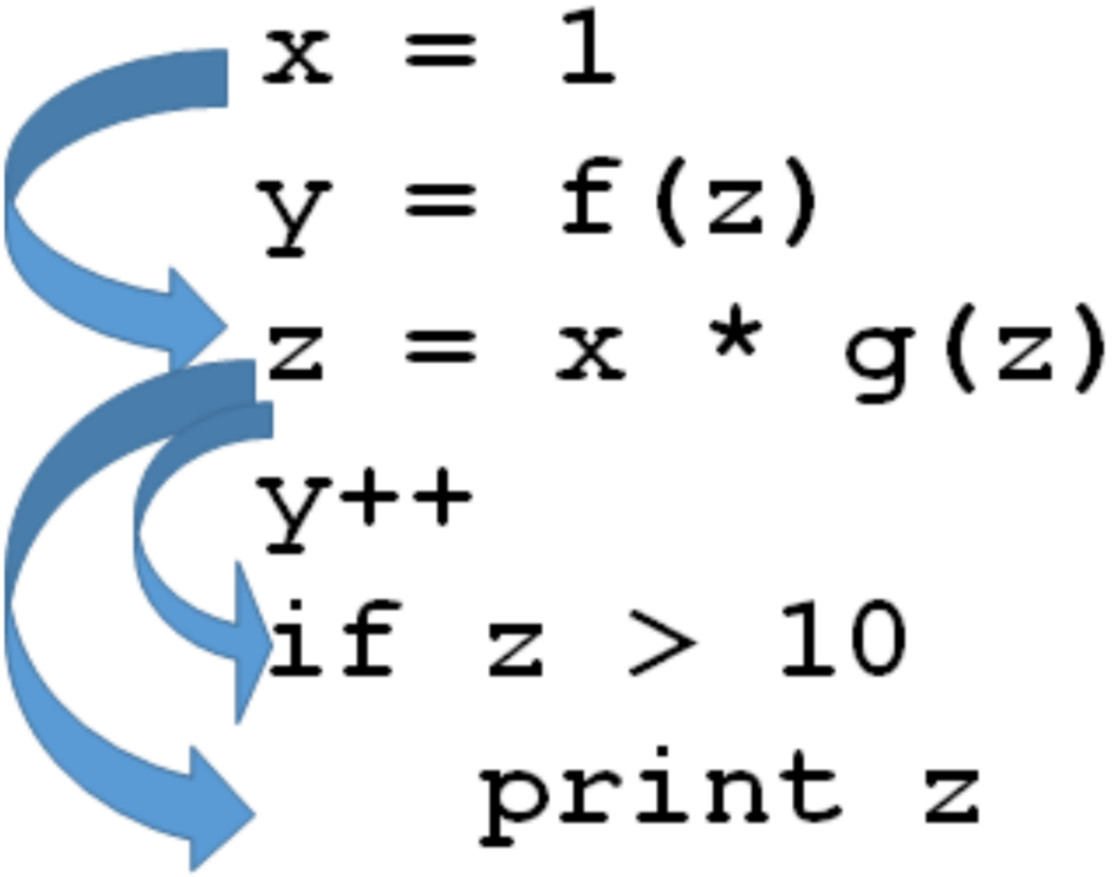
```
x = 1

print y
```

```
x = 1
y = f(z)
z = x * g(z)
y++
print z
```

=

```
x = 1
//y = f(z)
z = x * g(z)
//y++
print z
```

influence chains



```
x = 1
y = f(z)
z = x * g(z)
y++
if z > 10
   print z
```

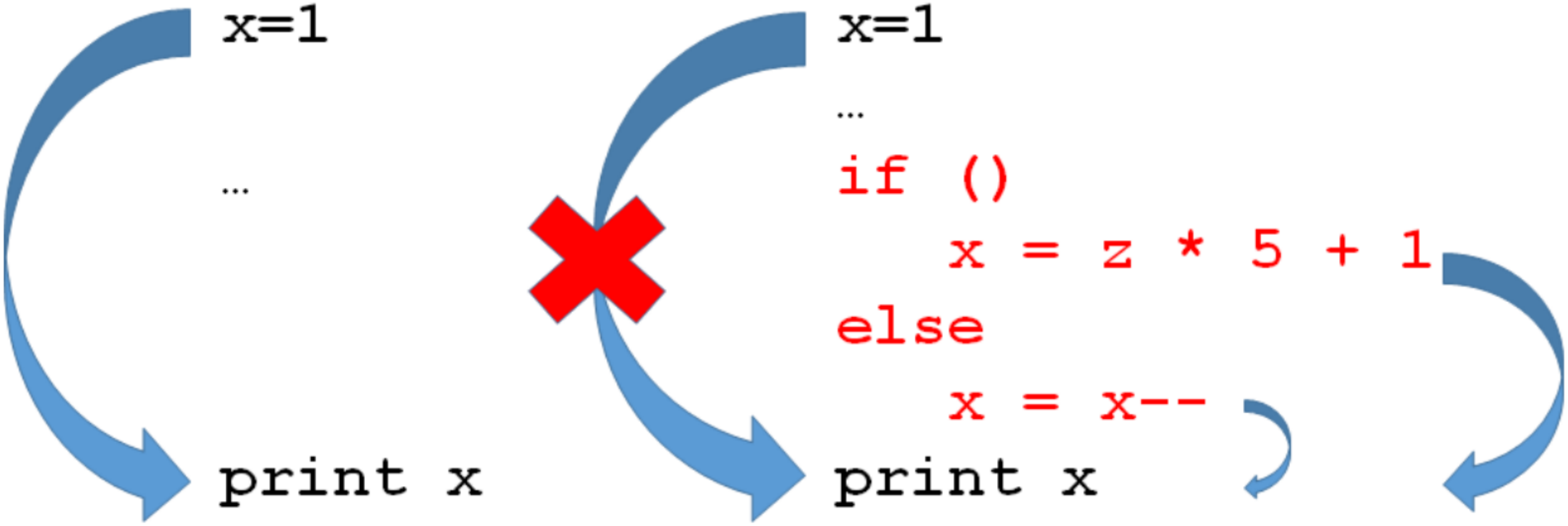
Influence debugging

reverse debugging along execution chains

advantages

missing case error

missing case error



```
x=1

...

print x
```

```
x=1
...
if ()
    x = z * 5 + 1
else
    x = x--
print x
```

Process of debugging

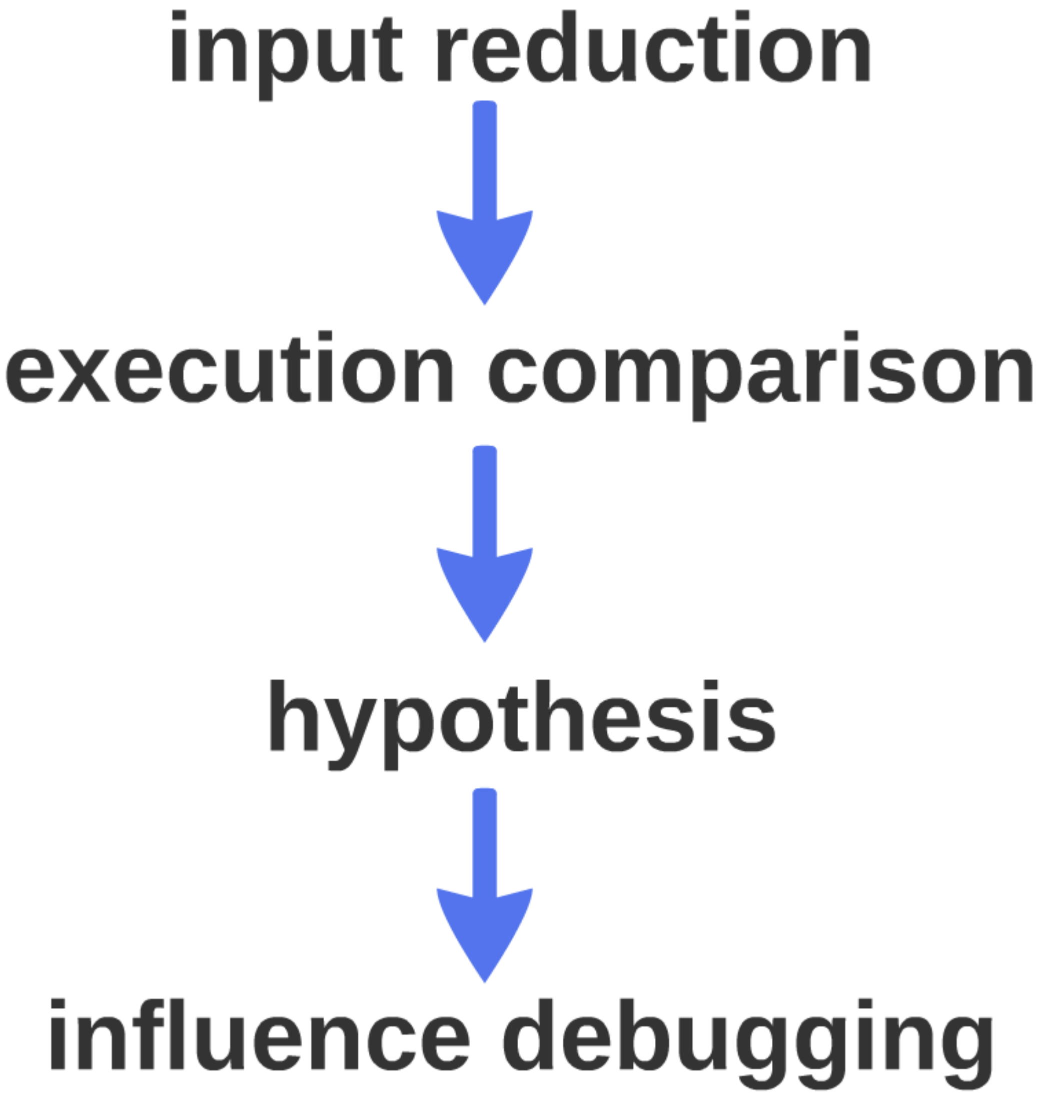
first step: code understanding

reducing executions steps

```
while (i <= j) {
  while (numbers[i] < pivot) {
    i++;
  }
  while (numbers[j] > pivot) {
    j--;
  }
  if (i <= j) {
     exchange(numbers, i, j);
  }
  i++;
  j--;
}
// Recursion
if (low < j)
  quicksort1(numbers, low, j);
if (i < high)
  quicksort1(numbers, i, high);
```

```
while (i <= j) {
    while (numbers[i] < pivot) {
      i++;
    }
    while (numbers[j] > pivot) {
      j--;
    }
    ~~if (i <= j) {~~
       ~~exchange(numbers, i, j);~~
    ~~}~~
    ~~i++;~~
    ~~j--;~~
  }
  // Recursion
  if (low < j)
    quicksort(numbers, low, j);
  if (i < high)
    quicksort(numbers, i, high);
```

active methods used together



input reduction

execution comparison

hypothesis

influence debugging

Feature location

Why code understanding is so important?

one third of the developers fix third party code

Feature location

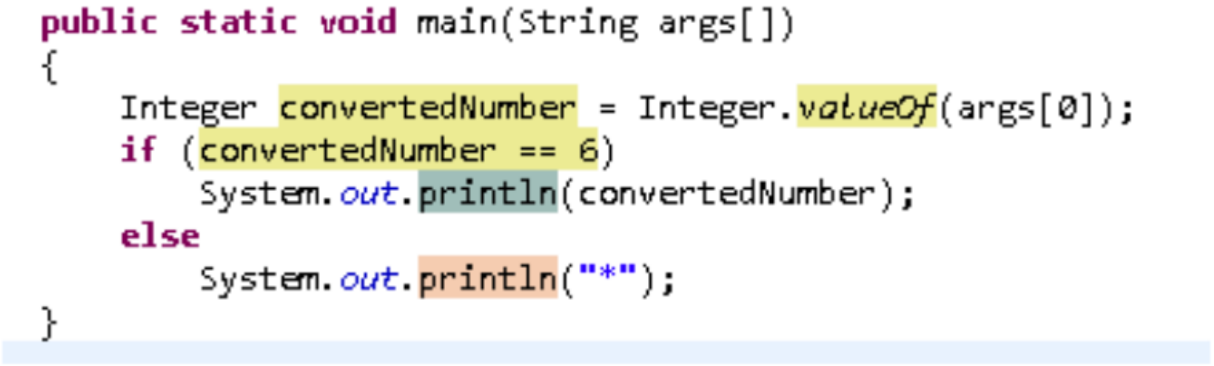
How to find the related code for the feature to modify?

execution comparison helps

how to find code to a feature

happy path

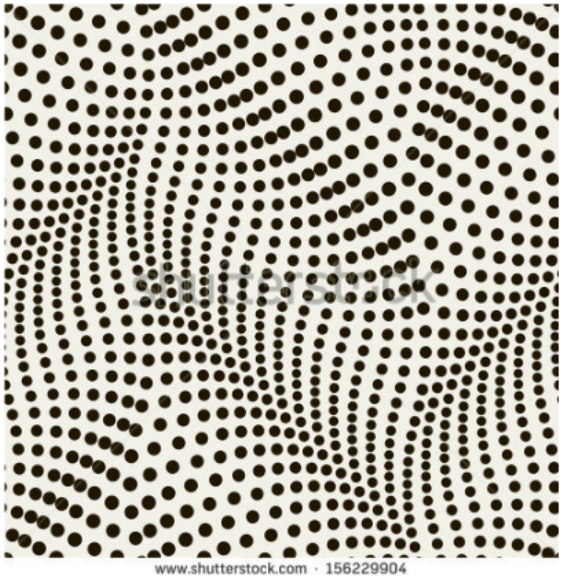
erroneous path



```
public static void main(String args[])
{
    Integer convertedNumber = Integer.valueOf(args[0]);
    if (convertedNumber == 6)
        System.out.println(convertedNumber);
    else
        System.out.println("*");
}
```

feature location in practice

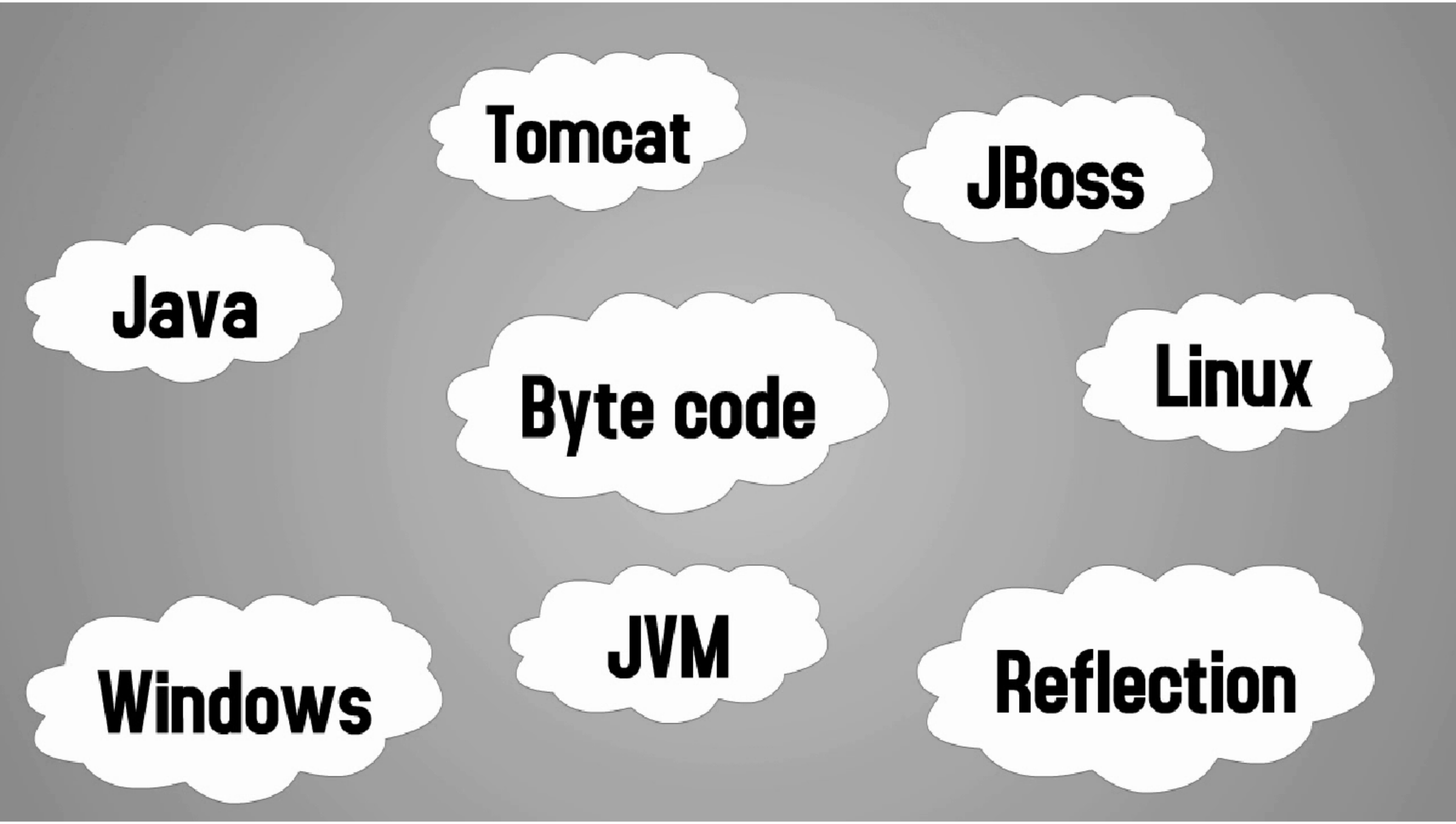
Let's try!
It's interesting and not tiresome



www.shutterstock.com · 156229904

jidebug

ACTIVE DEBUGGER



Tomcat

JBoss

Java

Byte code

Linux

Windows

JVM

Reflection

Jidebug features

records every program execution

recording can be started and stopped at any time

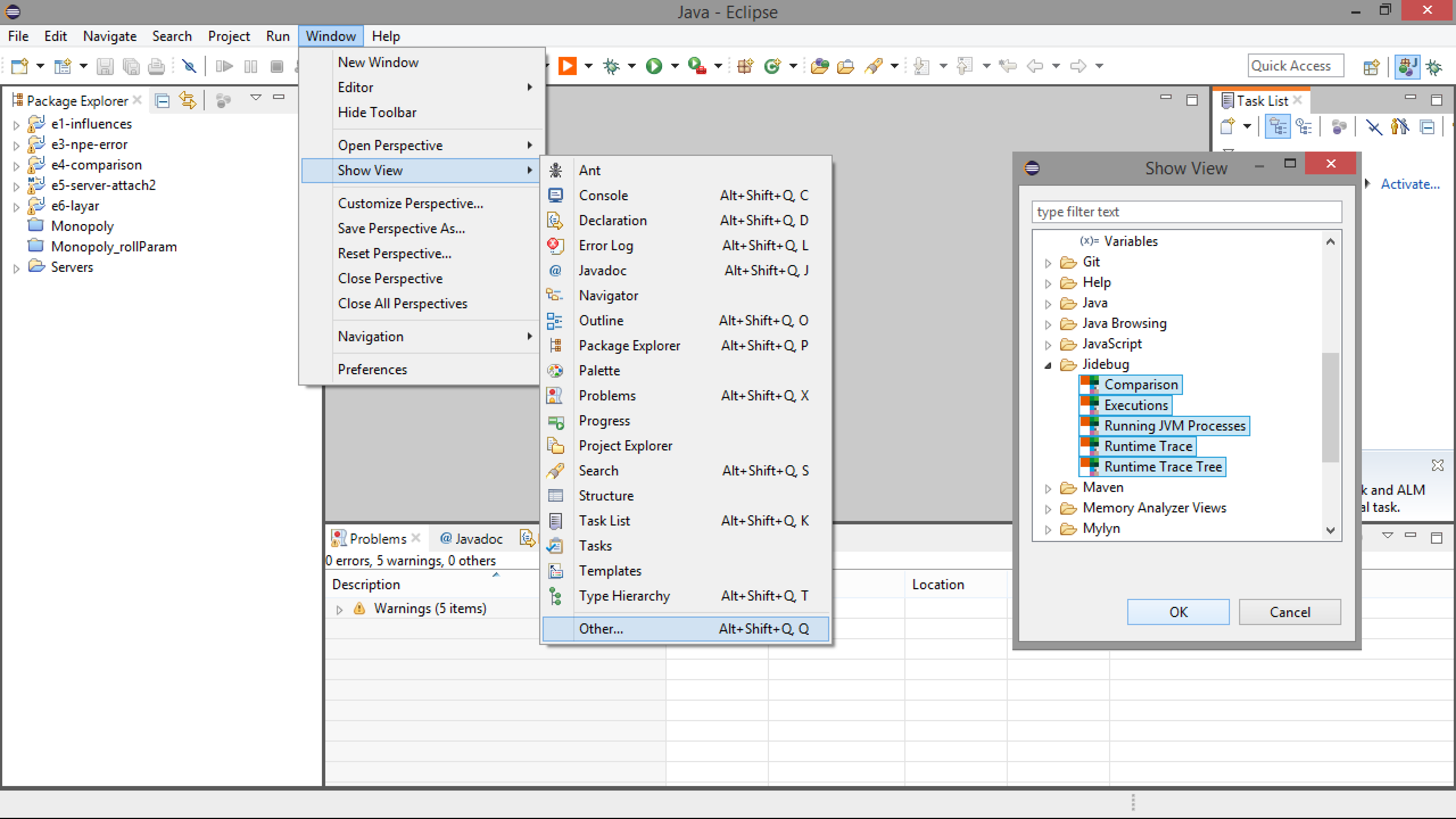
when stop it stores all relevant data

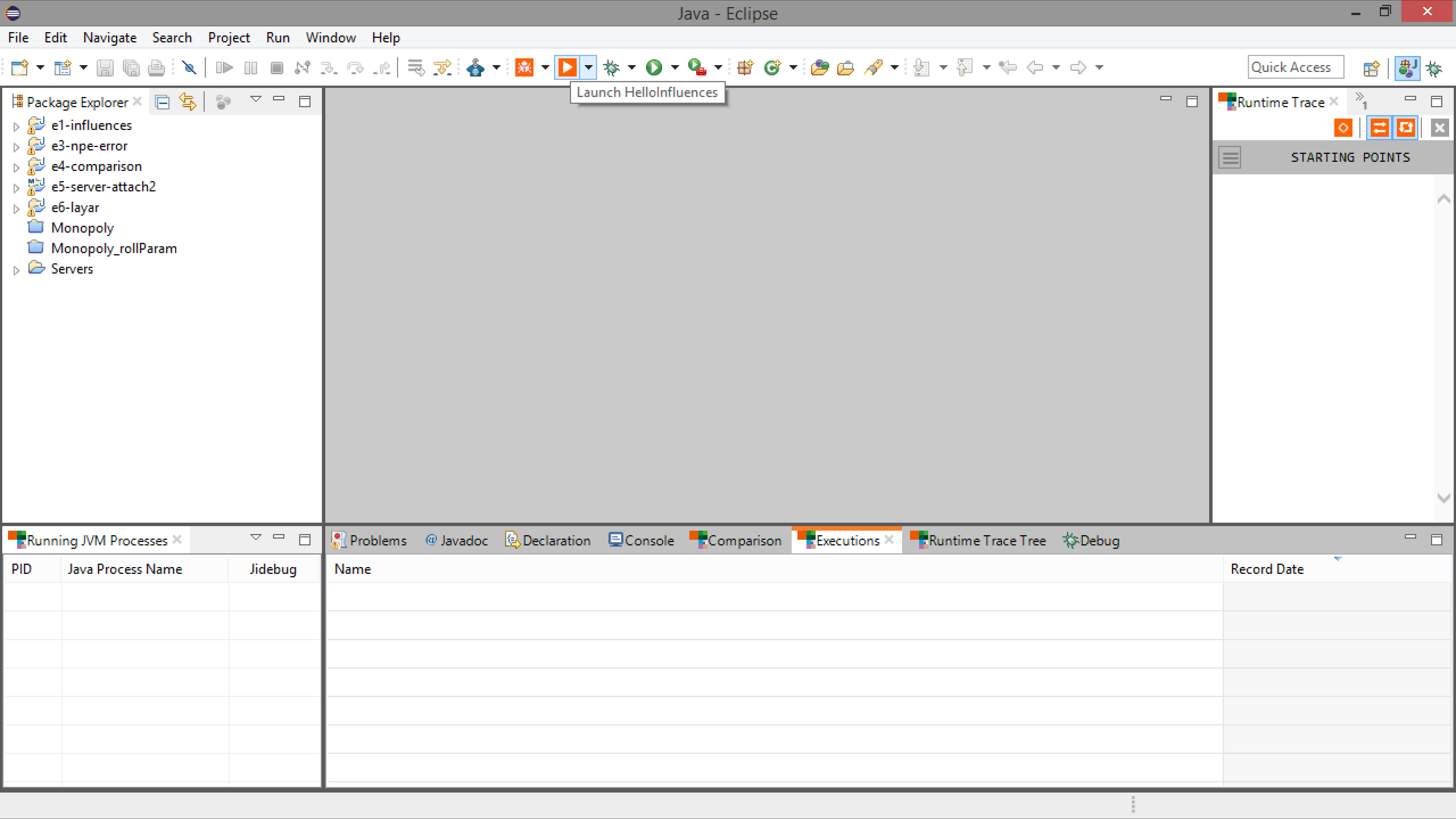
can be attached to any application already running on JVM

compares the executions and the differences are displayed

displays all the code parts which influence a selected variable

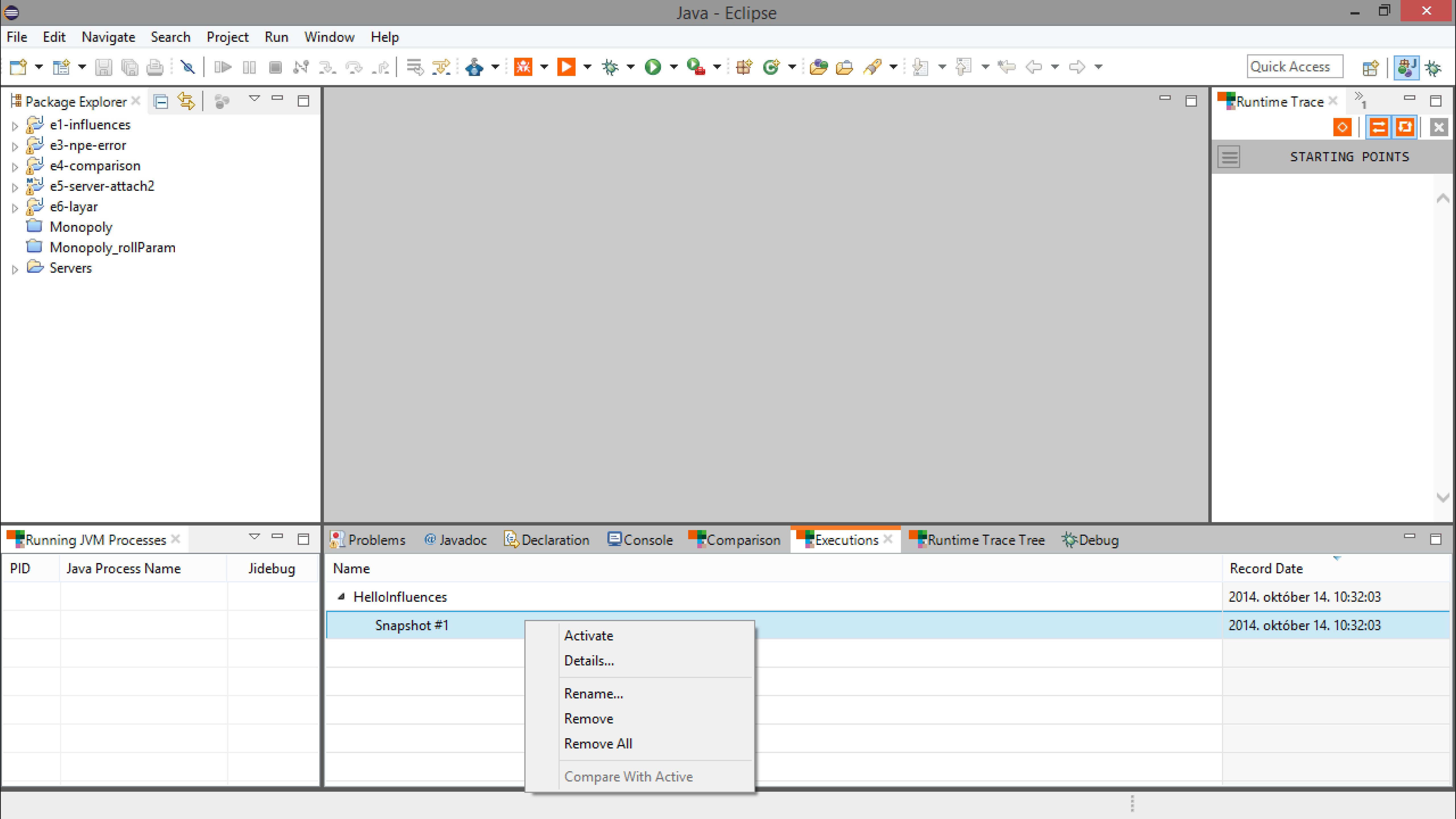
jidebug


Java - Eclipse
File
Edit
Navigate
Search
Project
Run
Window
Help
Quick Access
Package Explorer
e1-influences
e3-npe-error
e4-comparison
e5-server-attach2
e6-layar
Monopoly
Monopoly_rollParam
Servers
New Window
Editor
Hide Toolbar
Open Perspective
Show View
Customize Perspective...
Save Perspective As...
Reset Perspective...
Close Perspective
Close All Perspectives
Navigation
Preferences
Ant
Console Alt+Shift+Q, C
Declaration Alt+Shift+Q, D
Error Log Alt+Shift+Q, L
Javadoc Alt+Shift+Q, J
Navigator
Outline Alt+Shift+Q, O
Package Explorer Alt+Shift+Q, P
Palette
Problems Alt+Shift+Q, X
Progress
Project Explorer
Search Alt+Shift+Q, S
Structure
Task List Alt+Shift+Q, K
Tasks
Templates
Type Hierarchy Alt+Shift+Q, T
Other... Alt+Shift+Q, Q
Task List
Activate...
k and ALM
al task.
Show View
type filter text
(x)= Variables
Git
Help
Java
Java Browsing
JavaScript
Jidebug
Comparison
Executions
Running JVM Processes
Runtime Trace
Runtime Trace Tree
Maven
Memory Analyzer Views
Mylyn
OK
Cancel
Problems
@ Javadoc
0 errors, 5 warnings, 0 others
Description
Location
Warnings (5 items)



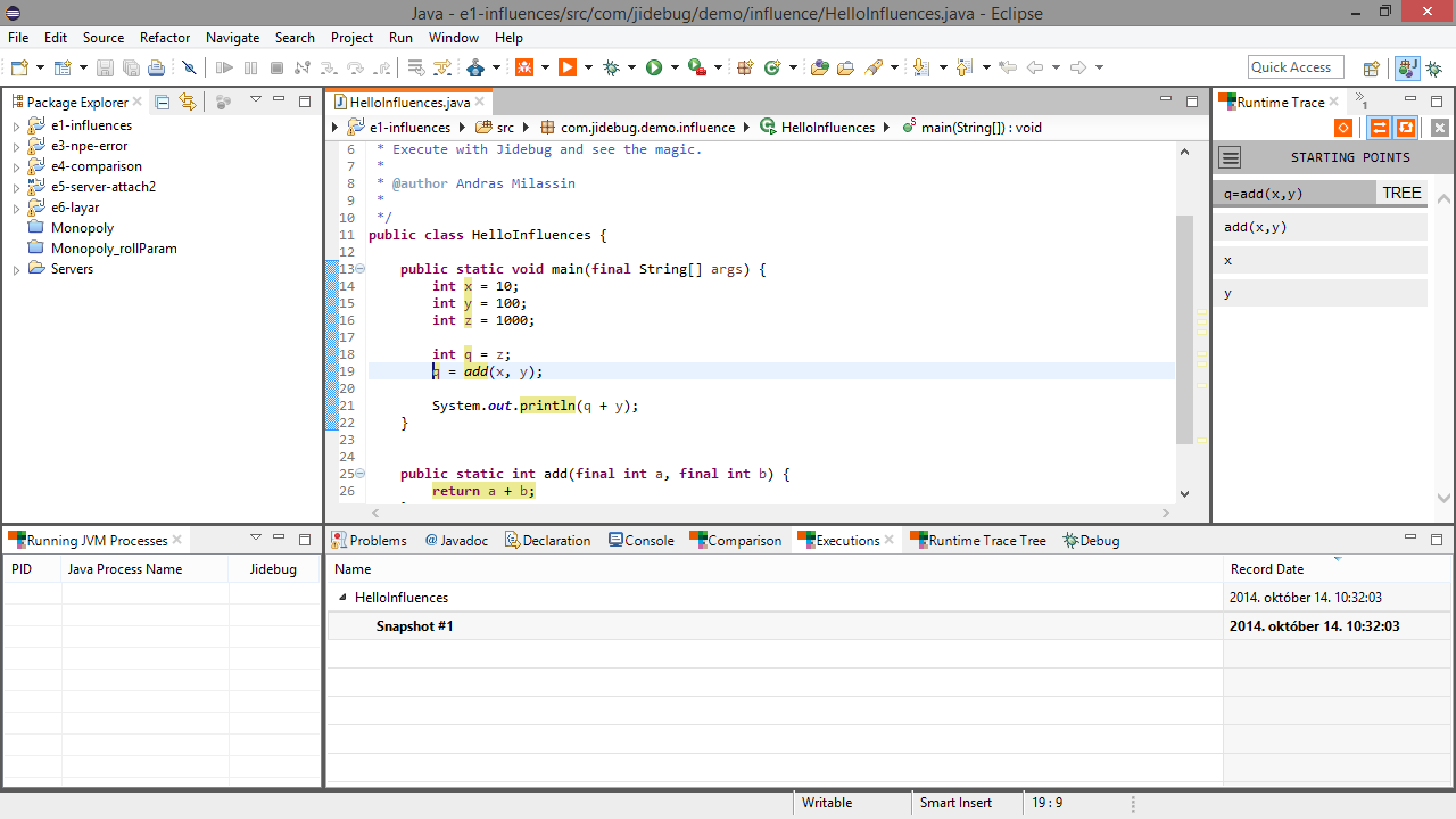
PID	Java Process Name	Jidebug

Name	Record Date



PID	Java Process Name	Jidebug

Name	Record Date
HelloInfluences	2014. október 14. 10:32:03
Snapshot #1	2014. október 14. 10:32:03



```
 * Execute with Jidebug and see the magic.
 *
 * @author Andras Milassin
 *
 */
public class HelloInfluences {

    public static void main(final String[] args) {
        int x = 10;
        int y = 100;
        int z = 1000;

        int q = z;
        q = add(x, y);

        System.out.println(q + y);
    }


    public static int add(final int a, final int b) {
        return a + b;
```

q=add(x,y)

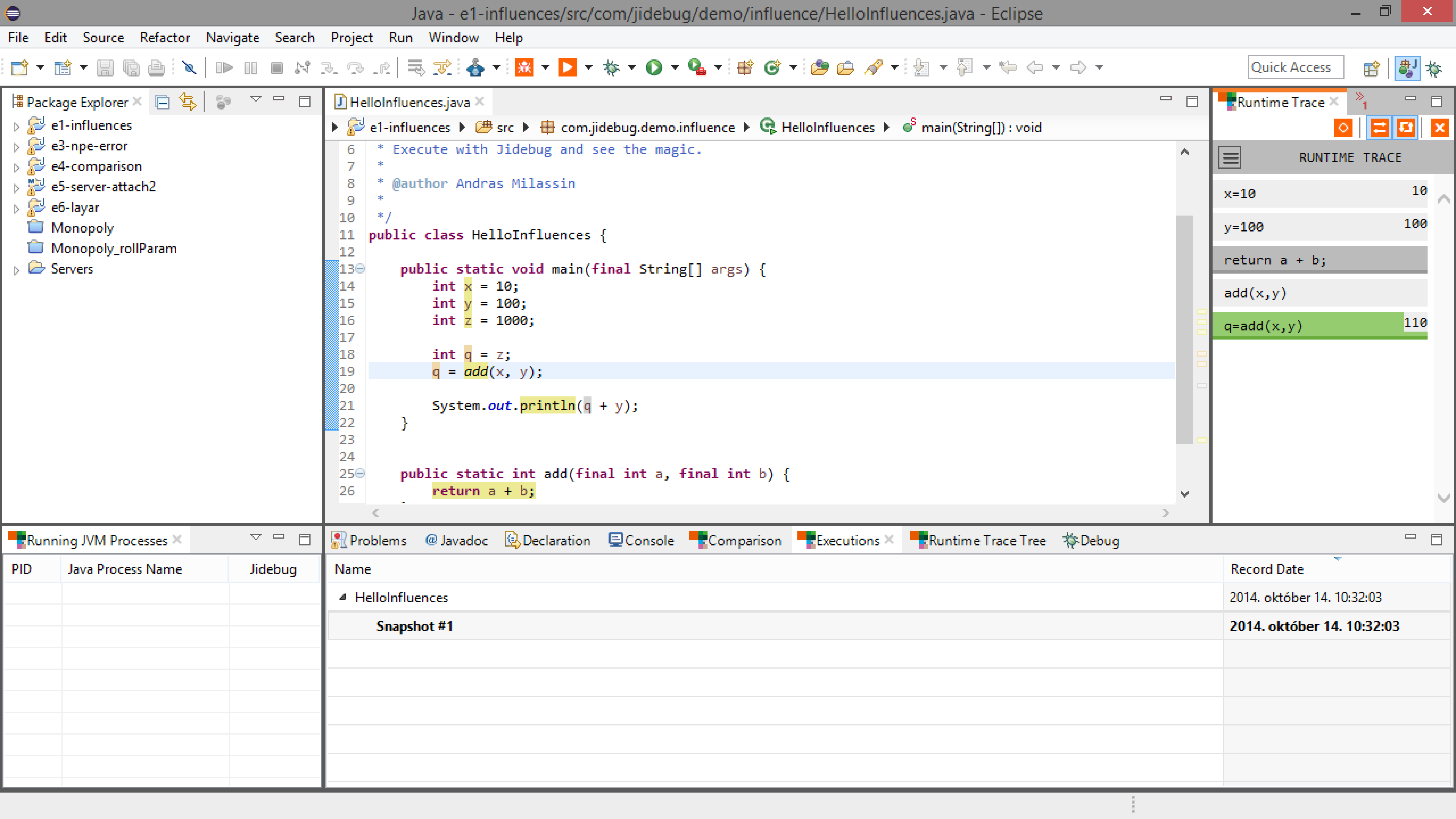
add(x,y)

x

y

PID	Java Process Name	Jidebug

Name	Record Date
HelloInfluences	2014. október 14. 10:32:03
Snapshot #1	**2014. október 14. 10:32:03**



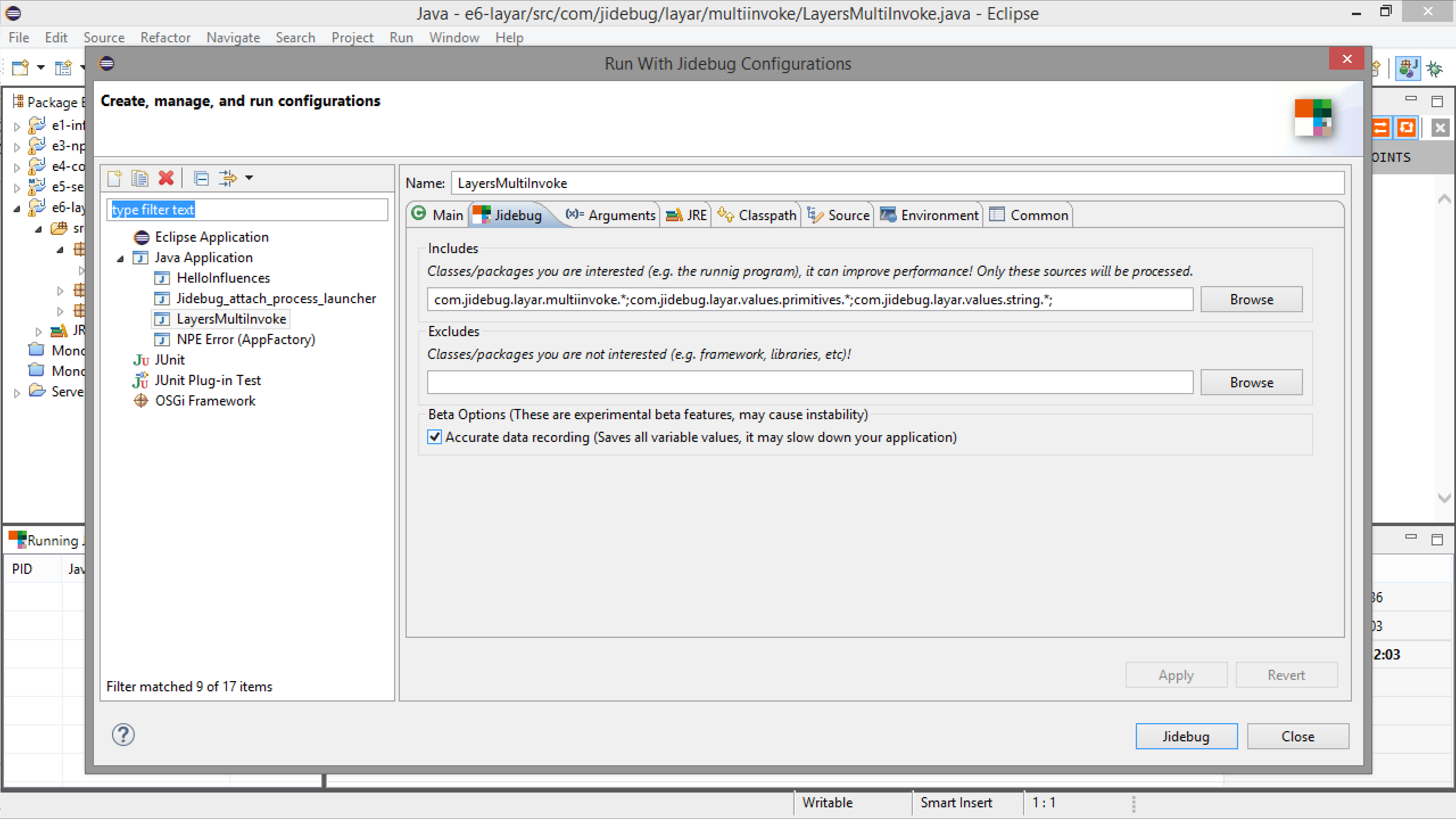
```
 * Execute with Jidebug and see the magic.
 *
 * @author Andras Milassin
 *
 */
public class HelloInfluences {

    public static void main(final String[] args) {
        int x = 10;
        int y = 100;
        int z = 1000;

        int q = z;
        q = add(x, y);

        System.out.println(q + y);
    }


    public static int add(final int a, final int b) {
        return a + b;
```

x=10	10
y=100	100
return a + b;	
add(x,y)	
q=add(x,y)	110

PID	Java Process Name	Jidebug

Name	Record Date
HelloInfluences	2014. október 14. 10:32:03
Snapshot #1	**2014. október 14. 10:32:03**

Tutorial 2: configurations & use precise data



Java - e6-layar/src/com/jidebug/layar/multiinvoke/LayersMultiInvoke.java - Eclipse

Run With Jidebug Configurations

Create, manage, and run configurations

type filter text

- Eclipse Application
- Java Application
  - HelloInfluences
  - Jidebug_attach_process_launcher
  - LayersMultiInvoke
  - NPE Error (AppFactory)
- JUnit
- JUnit Plug-in Test
- OSGi Framework

Filter matched 9 of 17 items

Name: LayersMultiInvoke

Includes

Classes/packages you are interested (e.g. the runnig program), it can improve performance! Only these sources will be processed.

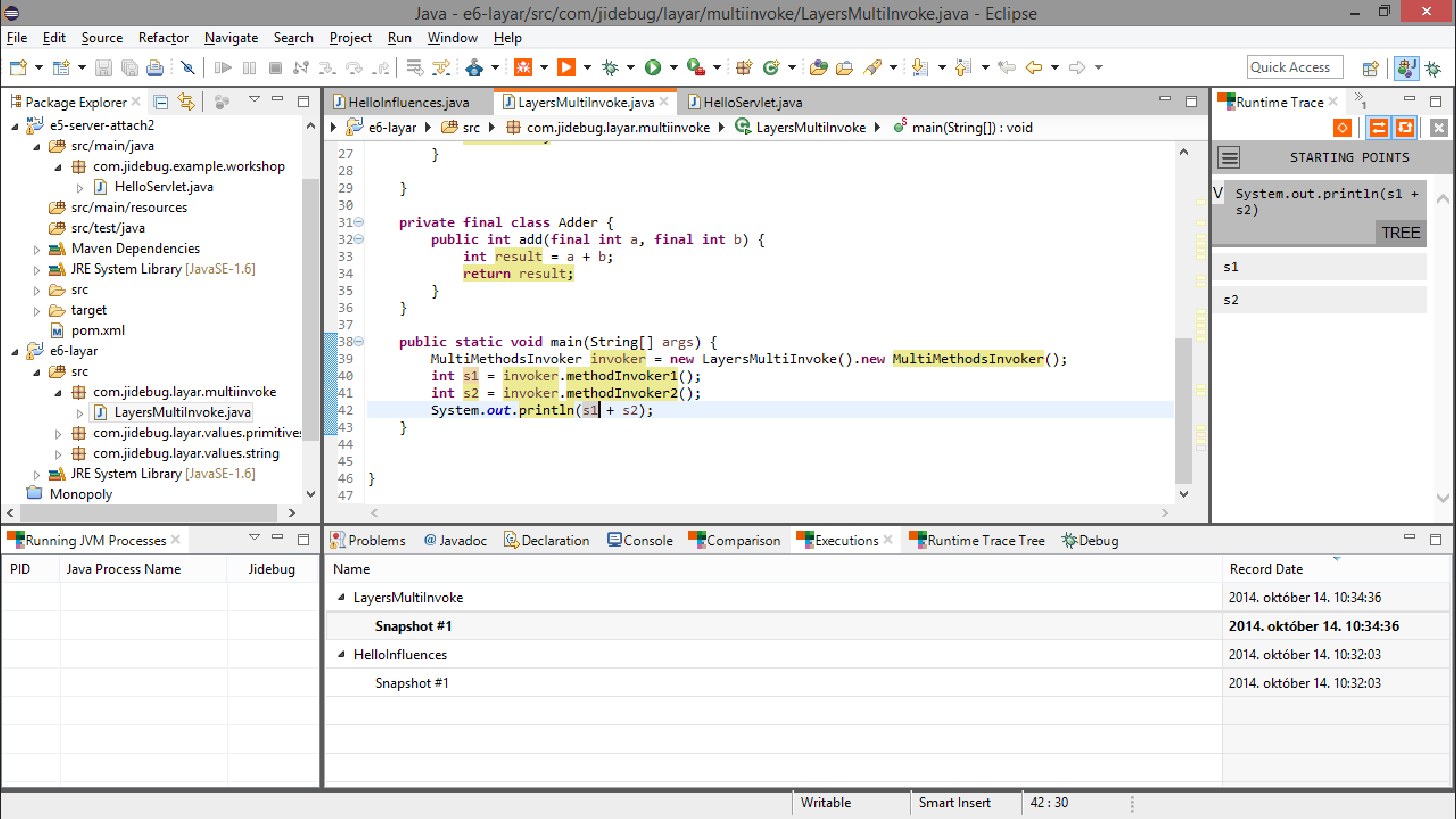
com.jidebug.layar.multiinvoke.*;com.jidebug.layar.values.primitives.*;com.jidebug.layar.values.string.*;

Excludes

Classes/packages you are not interested (e.g. framework, libraries, etc)!

Beta Options (These are experimental beta features, may cause instability)

☑ Accurate data recording (Saves all variable values, it may slow down your application)



Java - e6-layar/src/com/jidebug/layar/multiinvoke/LayersMultiInvoke.java - Eclipse

- e5-server-attach2
  - src/main/java
    - com.jidebug.example.workshop
      - HelloServlet.java
  - src/main/resources
  - src/test/java
  - Maven Dependencies
  - JRE System Library [JavaSE-1.6]
  - src
  - target
  - pom.xml
- e6-layar
  - src
    - com.jidebug.layar.multiinvoke
      - LayersMultiInvoke.java
    - com.jidebug.layar.values.primitives
    - com.jidebug.layar.values.string
  - JRE System Library [JavaSE-1.6]
- Monopoly

```
        }

    }

    private final class Adder {
        public int add(final int a, final int b) {
            int result = a + b;
            return result;
        }
    }

    public static void main(String[] args) {
        MultiMethodsInvoker invoker = new LayersMultiInvoke().new MultiMethodsInvoker();
        int s1 = invoker.methodInvoker1();
        int s2 = invoker.methodInvoker2();
        System.out.println(s1 + s2);
    }


}

```

STARTING POINTS

V System.out.println(s1 + s2)

TREE

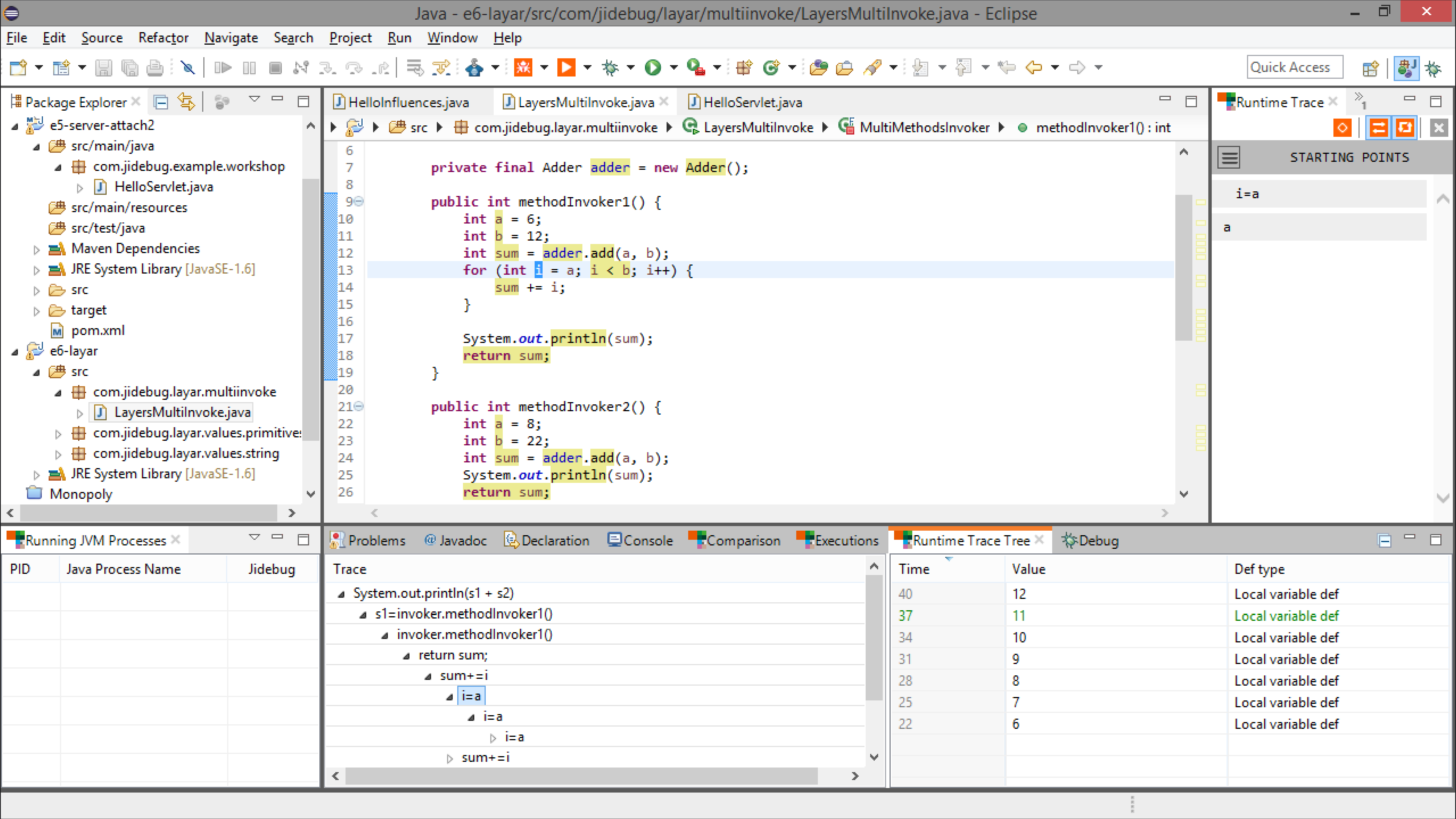
s1

s2

PID	Java Process Name	Jidebug

Name	Record Date
LayersMultiInvoke	2014. október 14. 10:34:36
Snapshot #1	**2014. október 14. 10:34:36**
HelloInfluences	2014. október 14. 10:32:03
Snapshot #1	2014. október 14. 10:32:03

Writable | Smart Insert | 42 : 30



```

        private final Adder adder = new Adder();

        public int methodInvoker1() {
            int a = 6;
            int b = 12;
            int sum = adder.add(a, b);
            for (int i = a; i < b; i++) {
                sum += i;
            }

            System.out.println(sum);
            return sum;
        }

        public int methodInvoker2() {
            int a = 8;
            int b = 22;
            int sum = adder.add(a, b);
            System.out.println(sum);
            return sum;
```

STARTING POINTS

i=a

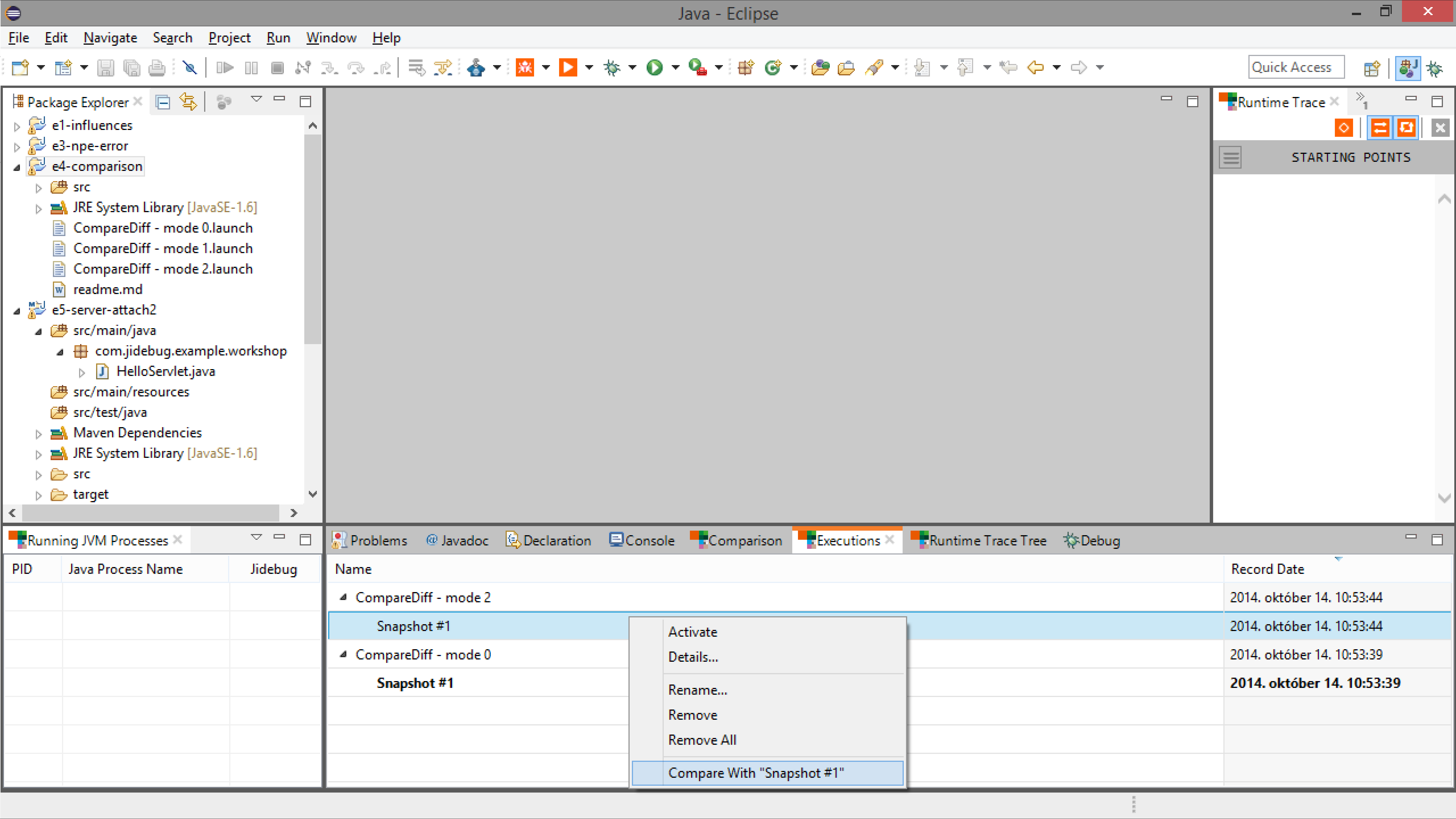
a

PID	Java Process Name	Jidebug

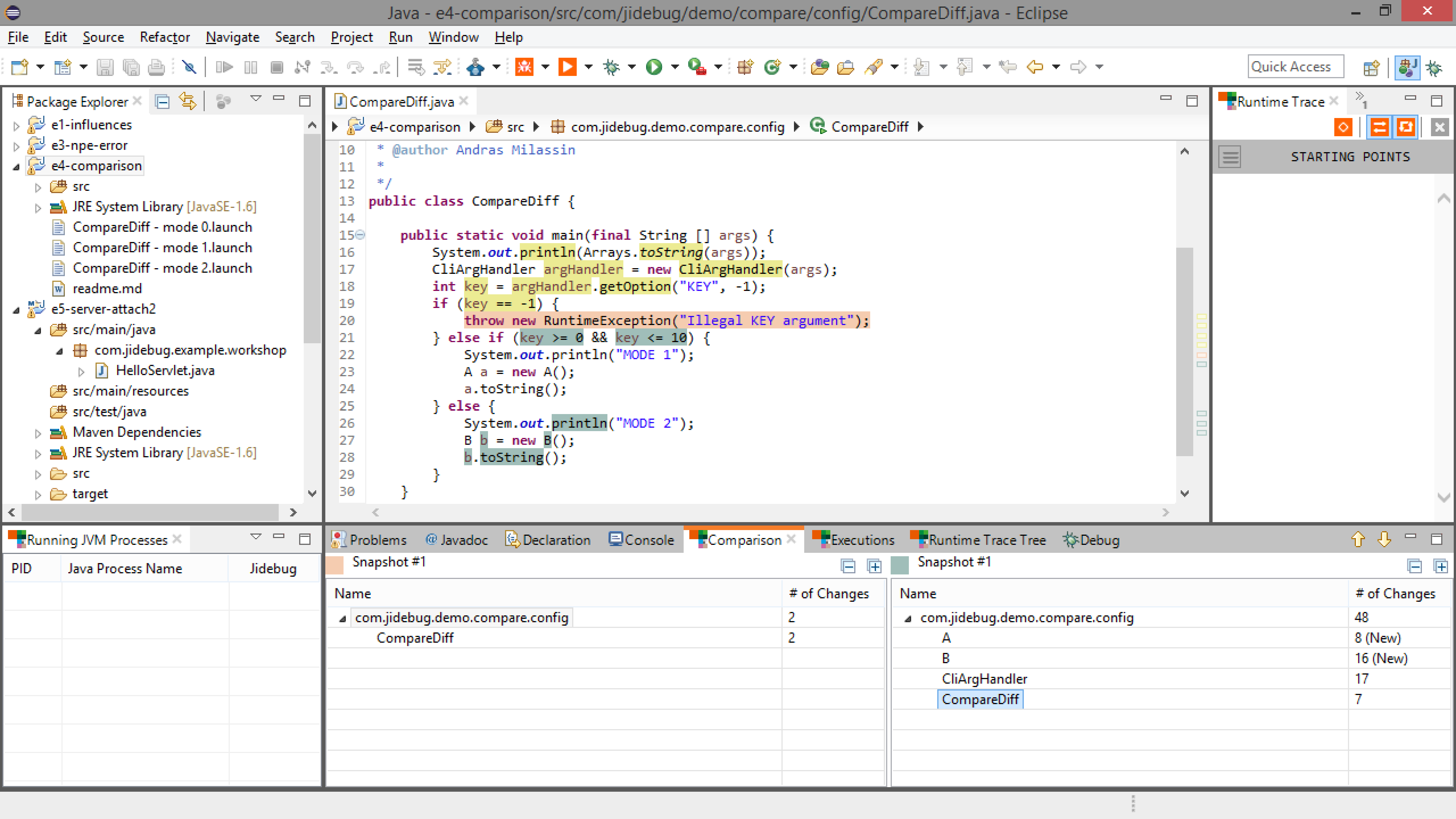
```
Trace
System.out.println(s1 + s2)
    s1=invoker.methodInvoker1()
        invoker.methodInvoker1()
            return sum;
                sum+=i
                    i=a
                        i=a
                            i=a
                    sum+=i
```

Time	Value	Def type
40	12	Local variable def
37	11	Local variable def
34	10	Local variable def
31	9	Local variable def
28	8	Local variable def
25	7	Local variable def
22	6	Local variable def

Tutorial 3: compare



Name	Record Date
CompareDiff - mode 2	2014. október 14. 10:53:44
Snapshot #1	2014. október 14. 10:53:44
CompareDiff - mode 0	2014. október 14. 10:53:39
Snapshot #1	**2014. október 14. 10:53:39**


Java - e4-comparison/src/com/jidebug/demo/compare/config/CompareDiff.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
e1-influences
e3-npe-error
e4-comparison
src
JRE System Library [JavaSE-1.6]
CompareDiff - mode 0.launch
CompareDiff - mode 1.launch
CompareDiff - mode 2.launch
readme.md
e5-server-attach2
src/main/java
com.jidebug.example.workshop
HelloServlet.java
src/main/resources
src/test/java
Maven Dependencies
JRE System Library [JavaSE-1.6]
src
target
CompareDiff.java
e4-comparison ▸ src ▸ com.jidebug.demo.compare.config ▸ CompareDiff ▸
Runtime Trace
STARTING POINTS
Running JVM Processes
Problems
Javadoc
Declaration
Console
Comparison
Executions
Runtime Trace Tree
Debug

```
 * @author Andras Milassin
 *
 */
public class CompareDiff {

    public static void main(final String [] args) {
        System.out.println(Arrays.toString(args));
        CliArgHandler argHandler = new CliArgHandler(args);
        int key = argHandler.getOption("KEY", -1);
        if (key == -1) {
            throw new RuntimeException("Illegal KEY argument");
        } else if (key >= 0 && key <= 10) {
            System.out.println("MODE 1");
            A a = new A();
            a.toString();
        } else {
            System.out.println("MODE 2");
            B b = new B();
            b.toString();
        }
    }
```

PID	Java Process Name	Jidebug

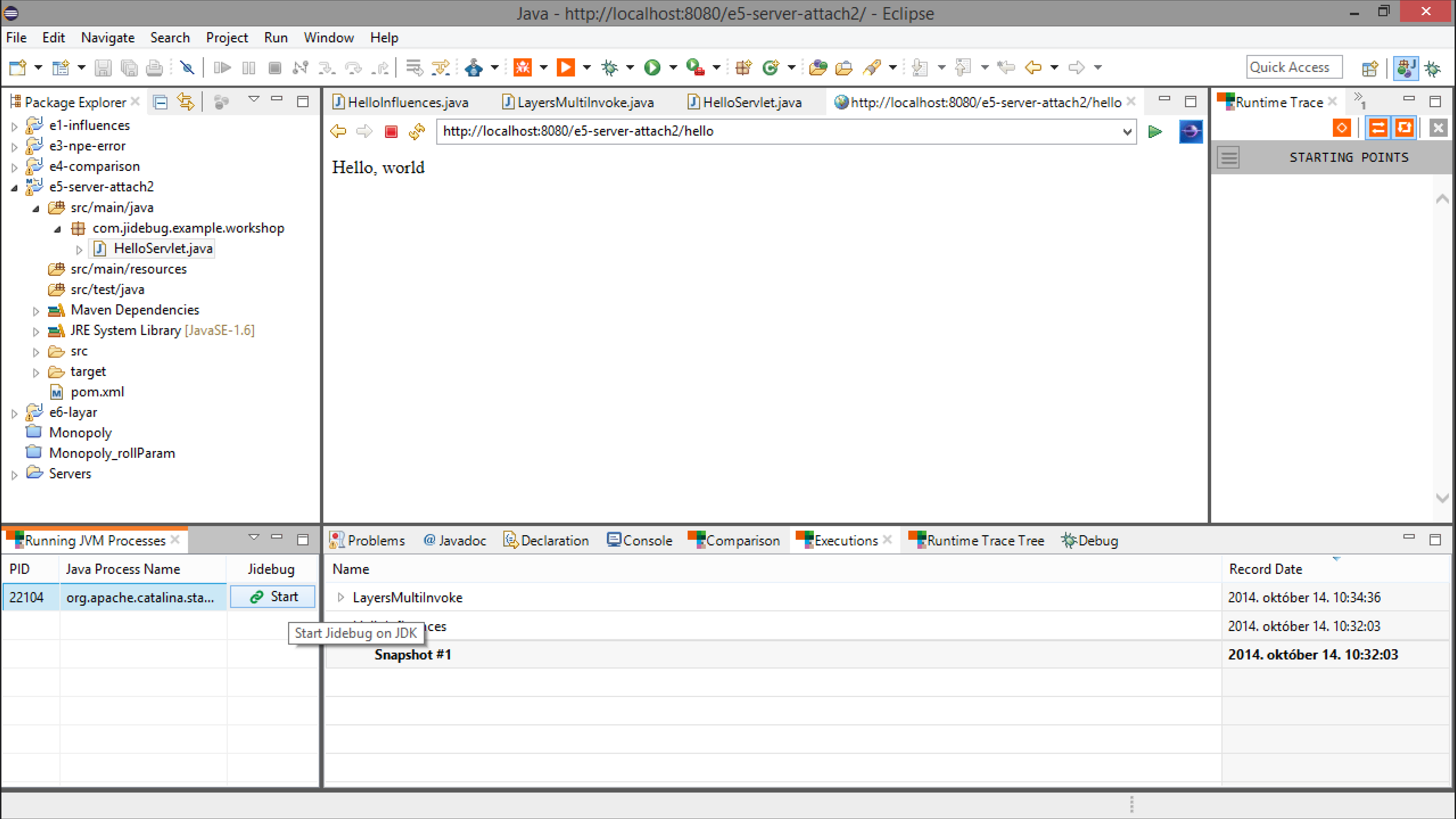
Snapshot #1

Name	# of Changes
com.jidebug.demo.compare.config	2
CompareDiff	2

Snapshot #1

Name	# of Changes
com.jidebug.demo.compare.config	48
A	8 (New)
B	16 (New)
CliArgHandler	17
CompareDiff	7

Tutorial 4:
attach to running process & recording



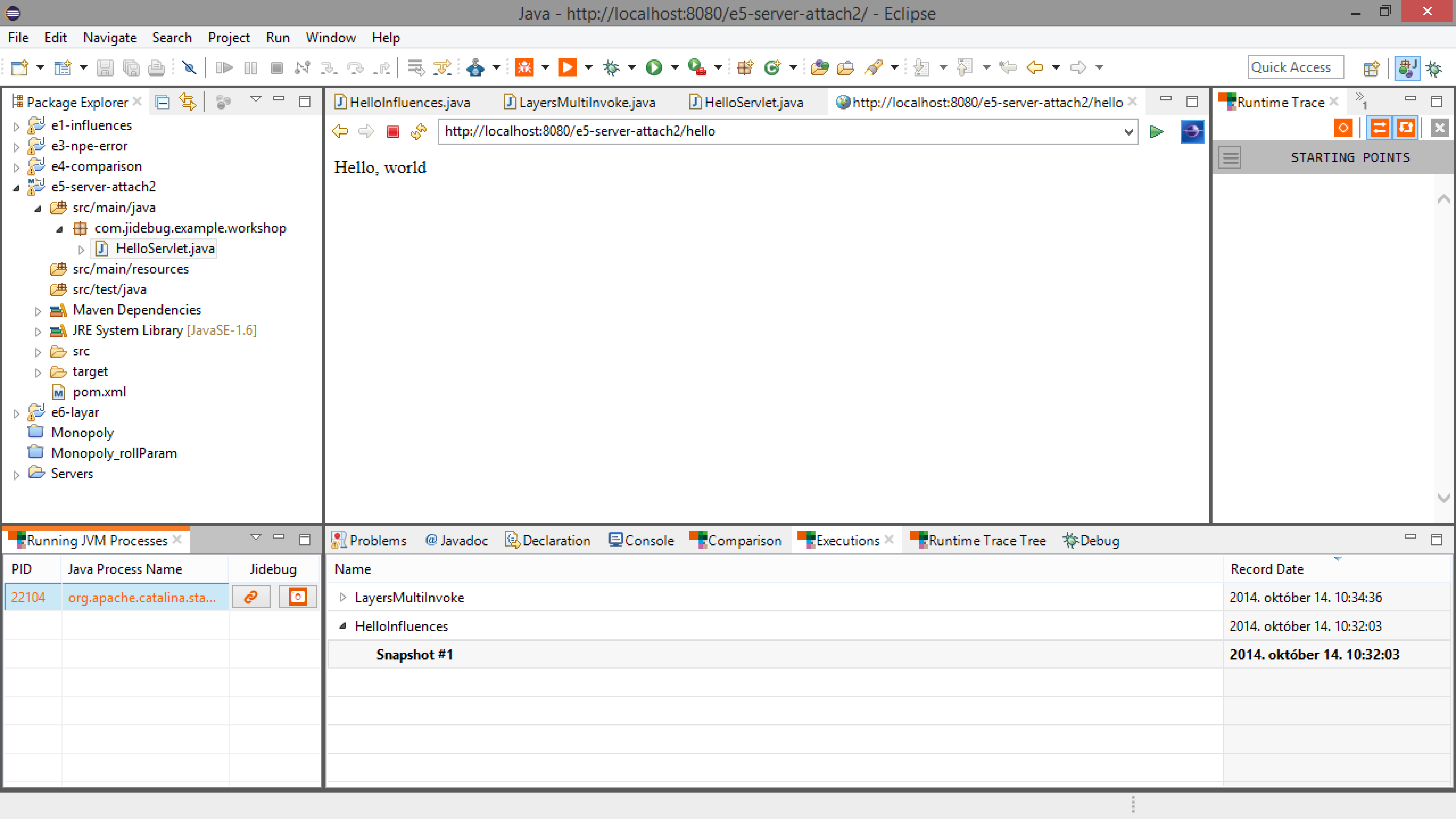
```
e1-influences
e3-npe-error
e4-comparison
e5-server-attach2
    src/main/java
        com.jidebug.example.workshop
            HelloServlet.java
    src/main/resources
    src/test/java
    Maven Dependencies
    JRE System Library [JavaSE-1.6]
    src
    target
    pom.xml
e6-layar
Monopoly
Monopoly_rollParam
Servers
```

http://localhost:8080/e5-server-attach2/hello

Hello, world

PID	Java Process Name	Jidebug
22104	org.apache.catalina.sta...	Start

Name	Record Date
LayersMultiInvoke	2014. október 14. 10:34:36
ces	2014. október 14. 10:32:03
Snapshot #1	**2014. október 14. 10:32:03**



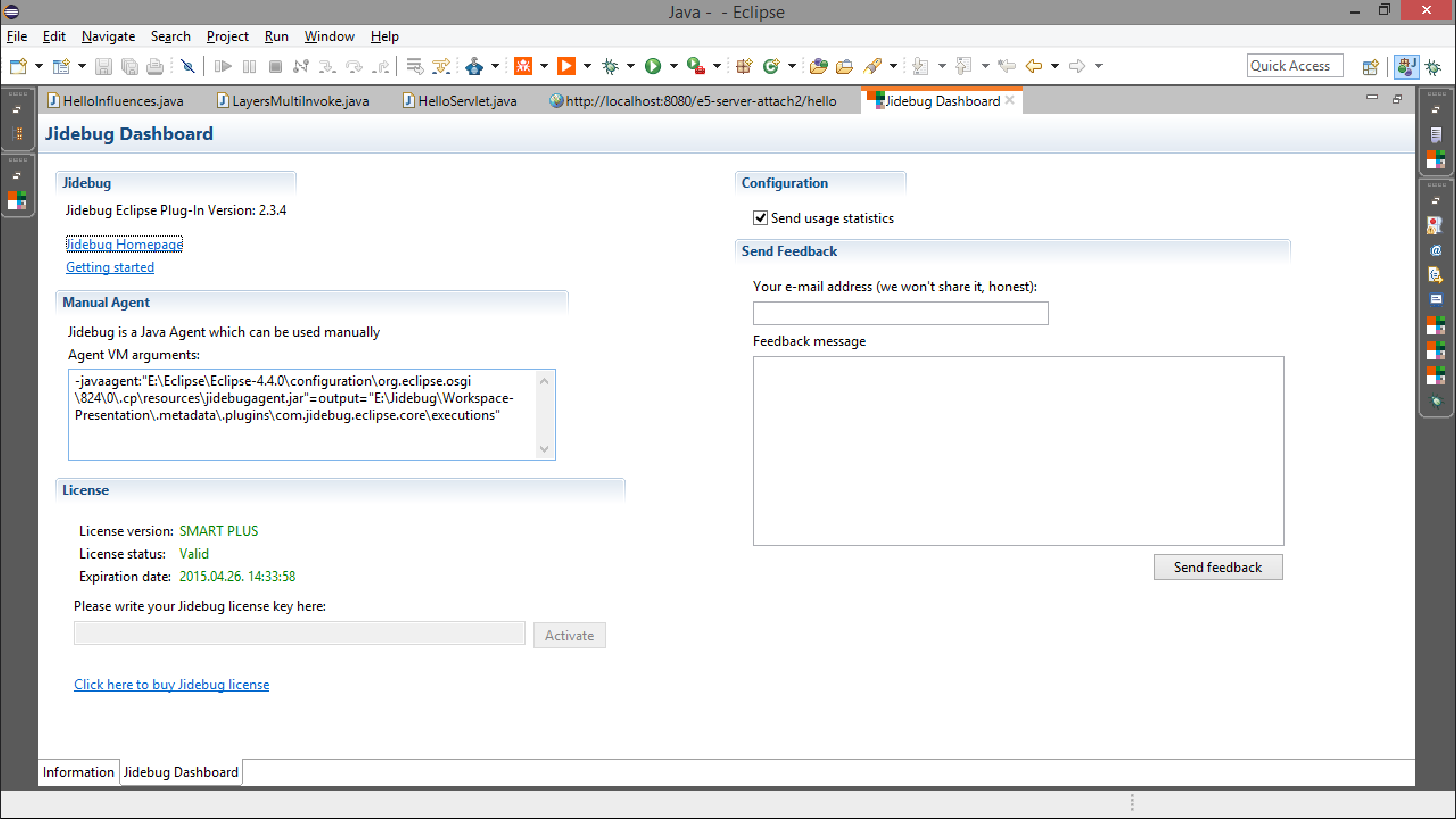
http://localhost:8080/e5-server-attach2/hello

Hello, world

PID	Java Process Name	Jidebug
22104	org.apache.catalina.sta...	

Name	Record Date
▷ LayersMultiInvoke	2014. október 14. 10:34:36
◢ HelloInfluences	2014. október 14. 10:32:03
Snapshot #1	**2014. október 14. 10:32:03**

Tutorial 6: dashboard



Jidebug Dashboard

Jidebug

Jidebug Eclipse Plug-In Version: 2.3.4

Jidebug Homepage

Getting started

Manual Agent

Jidebug is a Java Agent which can be used manually

Agent VM arguments:

```
-javaagent:"E:\Eclipse\Eclipse-4.4.0\configuration\org.eclipse.osgi
\824\0\.cp\resources\jidebugagent.jar"=output="E:\Jidebug\Workspace-
Presentation\.metadata\.plugins\com.jidebug.eclipse.core\executions"
```

License

License version: SMART PLUS

License status: Valid

Expiration date: 2015.04.26. 14:33:58

Please write your Jidebug license key here:

Activate

Click here to buy Jidebug license

Configuration

☑ Send usage statistics

Send Feedback

Your e-mail address (we won't share it, honest):

Feedback message

Send feedback

Try Jidebug

jidebug.com/download

jidebug.com

Istvan Forgacs

forgacs@jidebug.com

Andras Milassin

milassin@jidebug.com

@AMilassin

Jidebug is here

magic inside