COMP 249:
Object Oriented Programming II

Tutorial 4:

Inheritance and Polymorphism

Output Question 1

What is the output of the following program?

```
class Base {
        public void print() {
                System.out.println("Base");
        }
}
class Derived extends Base {
        public void print() {
                System.out.println("Derived");
        }
}
class Main{
        public static void doPrint( Base o ) {
                o.print();
        }
        public static void main(String[] args) {
                Base x = new Base();
                Base y = new Derived();
                Derived z = new Derived();
                doPrint(x);
                doPrint(y);
                doPrint(z);
        }
}
```

Output Question 2

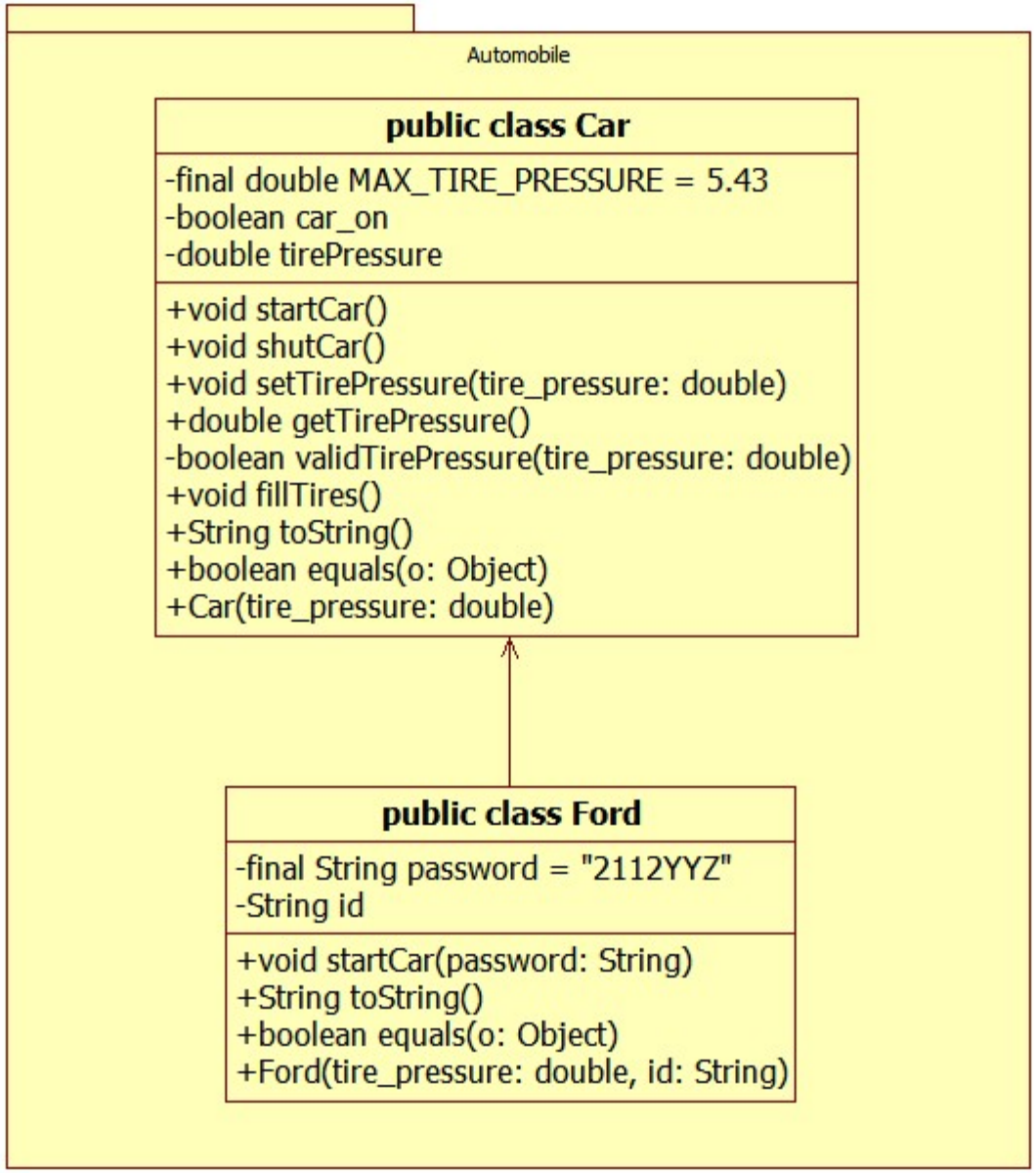
What are the errors in the following program and how they can be fixed ?

```
public class A {
    private int a = 100;
    public void setA( int value) {
        a = value;
}    public int getA() {
        return a;
    }
}
```

```
public class OOPExercises {
    public static void main(String[] args)
{
        A objA = new A();
        System.out.println("in main(): ");
        System.out.println("objA.a = "+objA.a);
        objA.a = 222;
    }
}
```

Programming Question

- Implement the UML diagram on the right.
- Both Classes in package “Automobile”, Driver.java should be in a separate (default) package.
- *Privacy Legend:*
  - *+ means public*
  - *~ means package*
  - *# means protected*
  - *- means private*



Programming Question

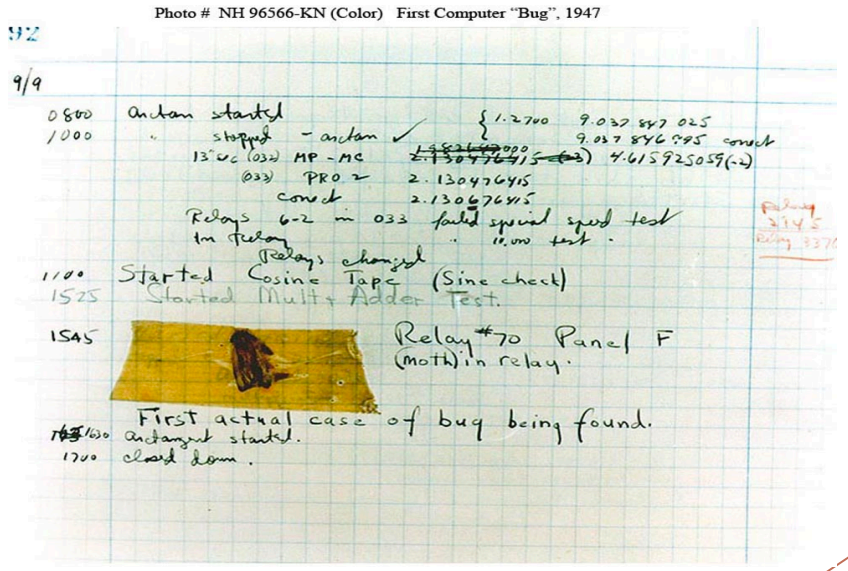
- startCar(…) of Ford class should overload that of the Car class, which allows the user to optionally enter a password that would unlock special features not implemented in this program.
  - Print an appropriate message if the password is correct.
  - Method should call the startCar() of the superclass regardless of password validity.
- Overriding toString() from the Ford class should use that of its superclass.
- Proper implementation of the equals method consists of handling null references and object-attribute comparison.

What is a software Bug ?

- A defect in a computer program causing it to malfunction
- Reasons:
  - Insufficient logic or erroneous logic
  - Design flaws
  - Hardware failures
  - Etc.

What is a software Bug ? Cont'd.

- First computer bug was in fact an actual Bug ☺
- 1945 at Harvard, a Moth trapped between two electrical relays of the Mark II Aiken Relay Calculator caused the whole machine to shut down.



Some famous Bugs

- The European Space Agency's Ariane 5 Flight 501 was destroyed 40 seconds after takeoff (June 4, 1996) due to a bug in the on-board guidance software.
- NASA's Spirit rover became unresponsive on January 21, 2004, a few weeks after landing on Mars due to accumulation of too many files in the rover's flash memory.
- The 2003 North America blackout was triggered by a local outage that went undetected due to a race condition in General Electric Energy's XA/21 monitoring software.
- Smart ship USS Yorktown was left dead in the water in 1997 for nearly 3 hours after a divide by zero error.

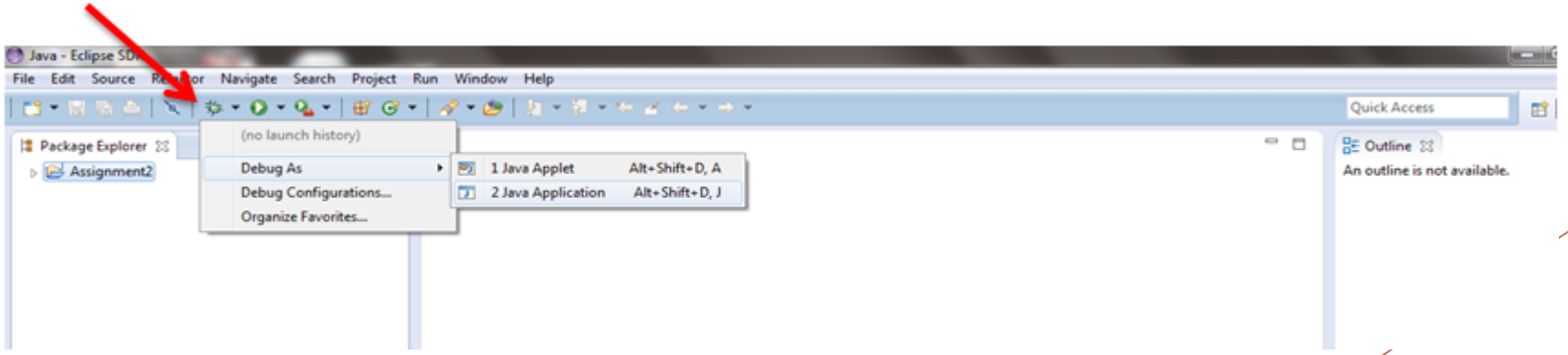
What is software Debugging?

- The process of investigating and fixing defects / bugs within a computer program.

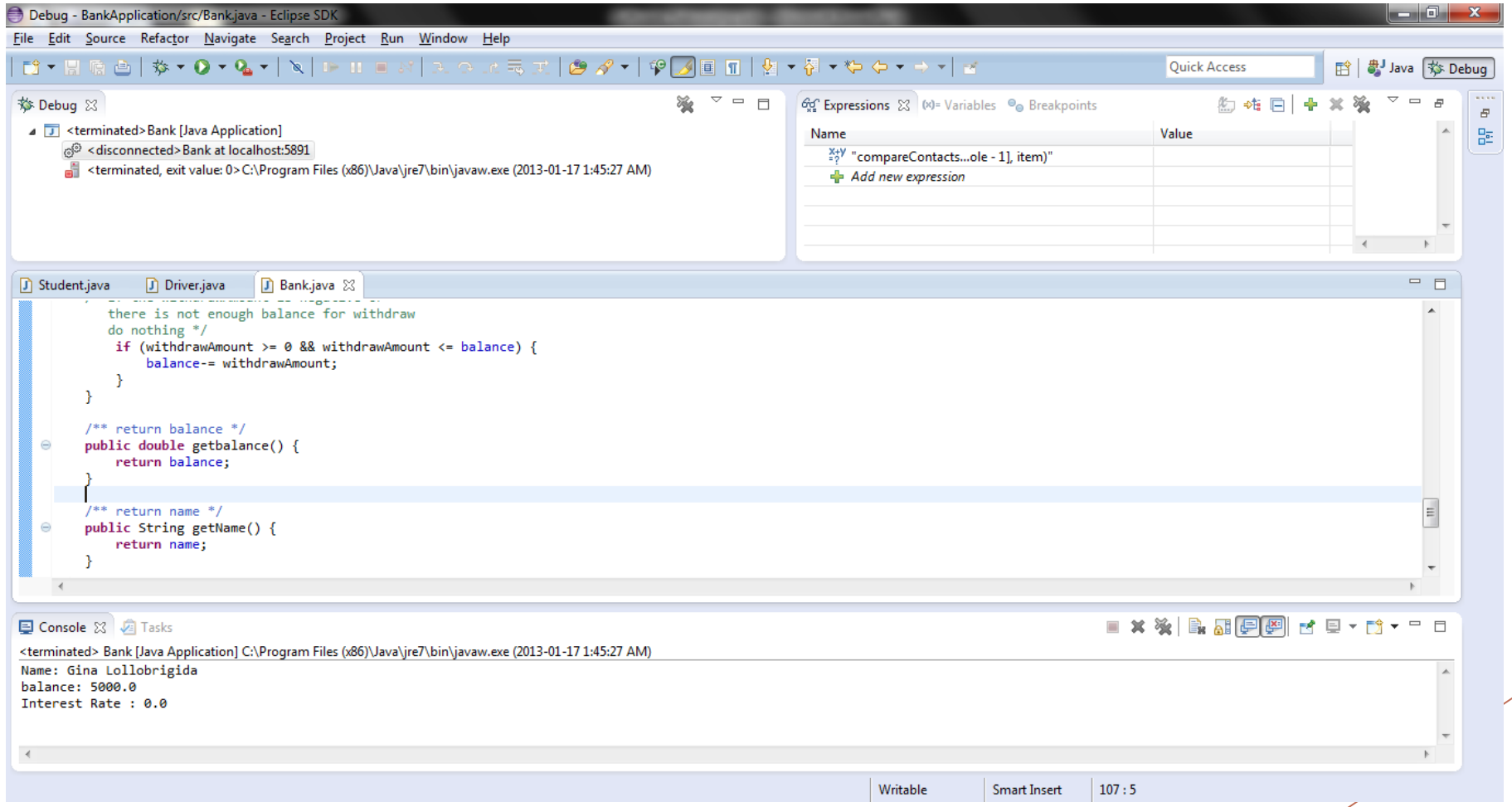
- Methods of debugging
  - **printf** debugging
  - Log scraping
  - Post-mortem debugging
    - Debugging of the program after it has already crashed.
    - Create core dumps/crash dumps
    - Analyse using various tools : WinDbg , gdb
  - built-in debugging features in IDE Platforms, Visual Studio.NET, Eclipse, NetBeans etc.

Debug using Eclipse

- Debugging support on Eclipse
  - Provide and execution mode ‘Debug’
  - Lots of features to investigate the program behaviour while it’s executing.



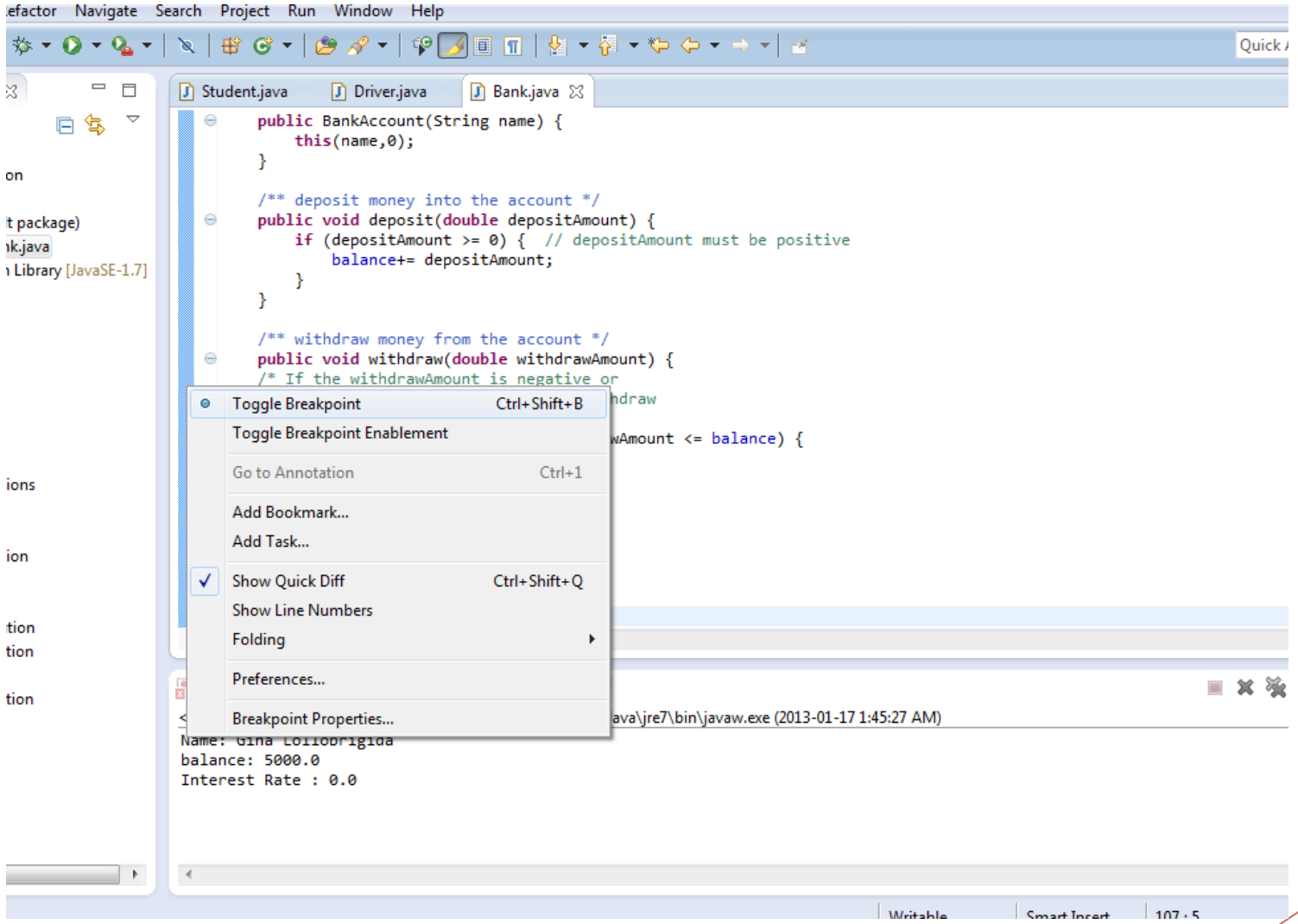
Eclipse Debug view



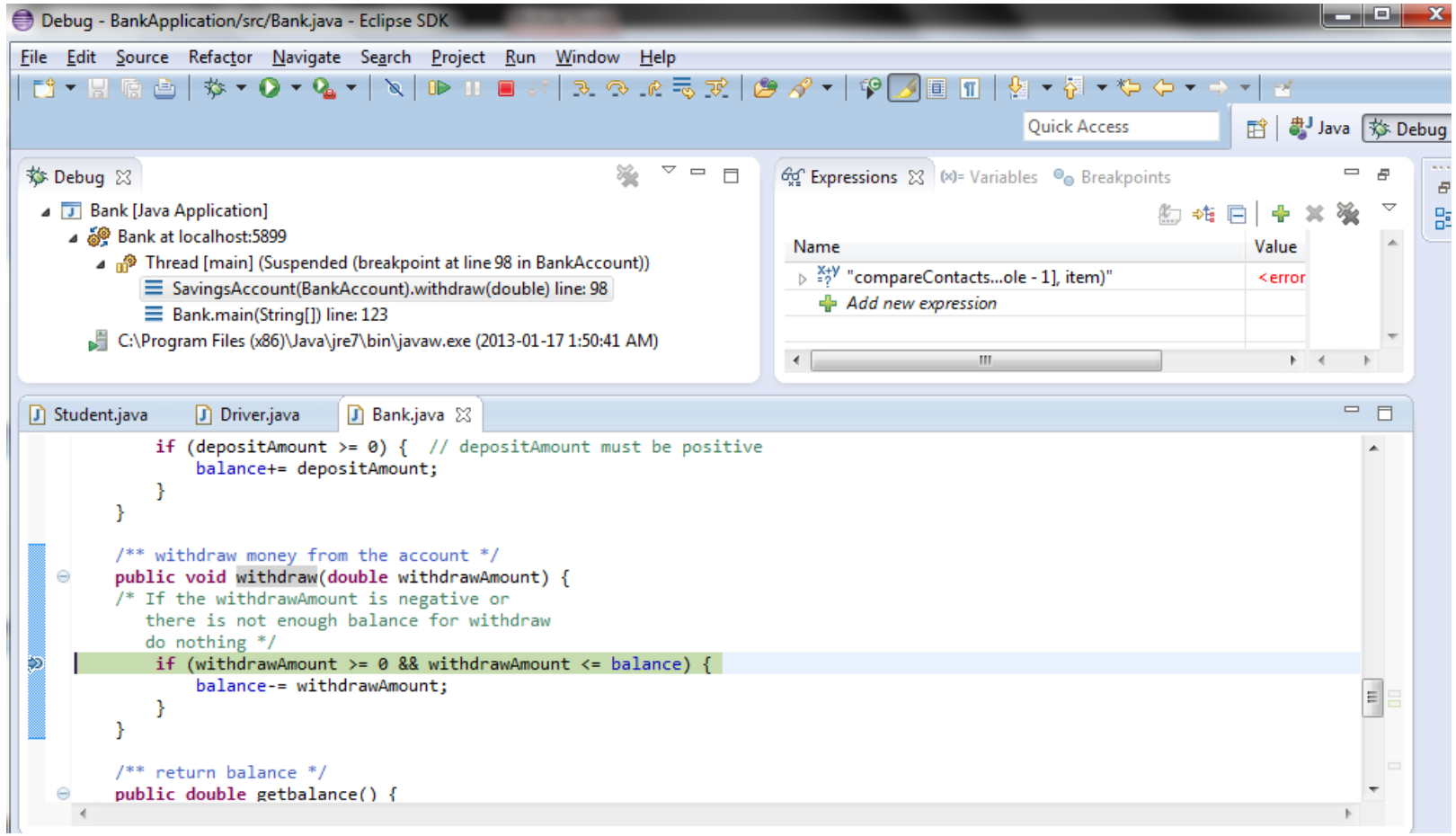
Eclipse Debug Features: Breakpoints

- Breakpoints
  - Set a particular breakpoint inside the code.
  - When executed in Debug mode, the execution get suspended at the breakpoint.
  - Gives the facility to investigate current execution status of the program.

Setting a breakpoint in Eclipse

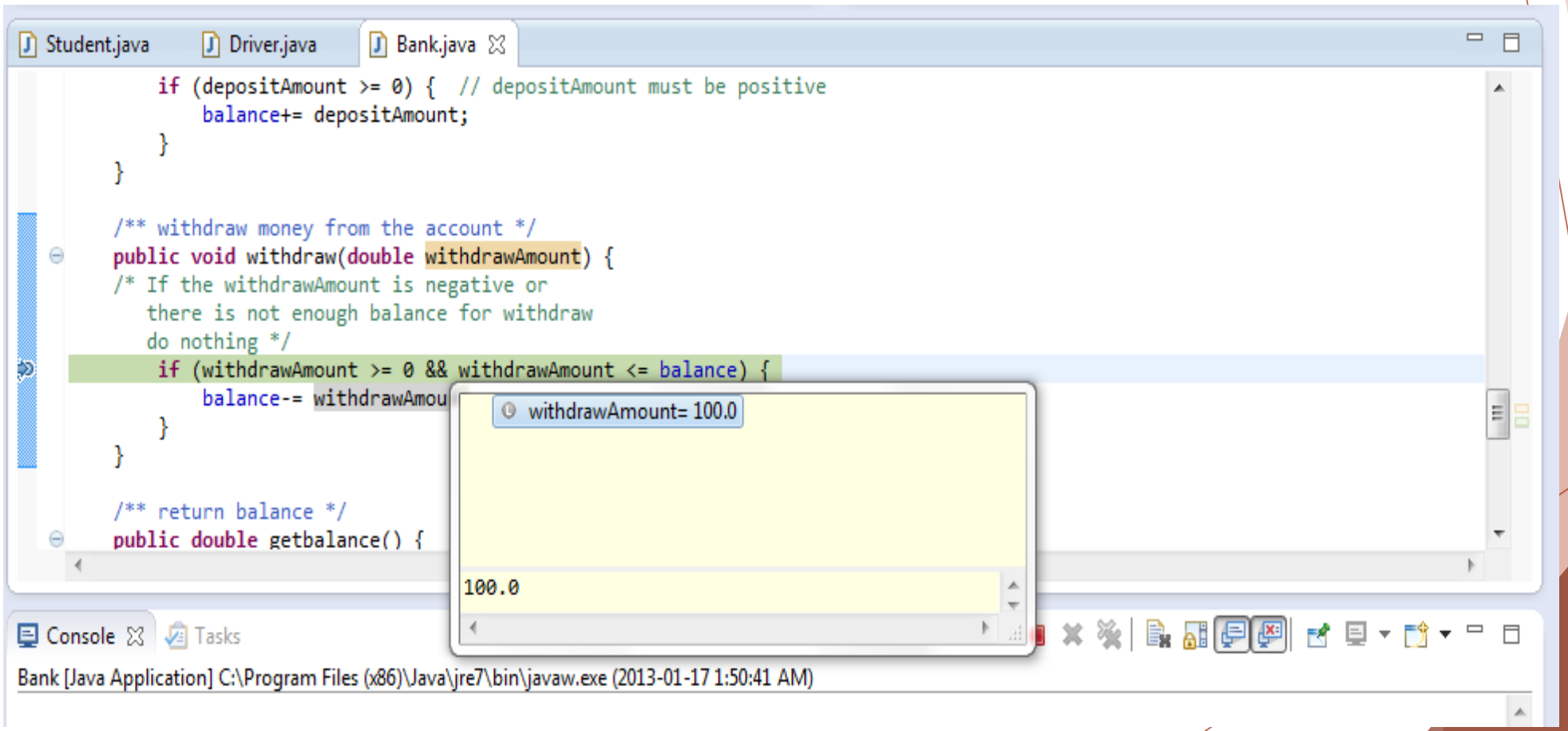


Eclipse Debug Features: Breakpoints cont'd.



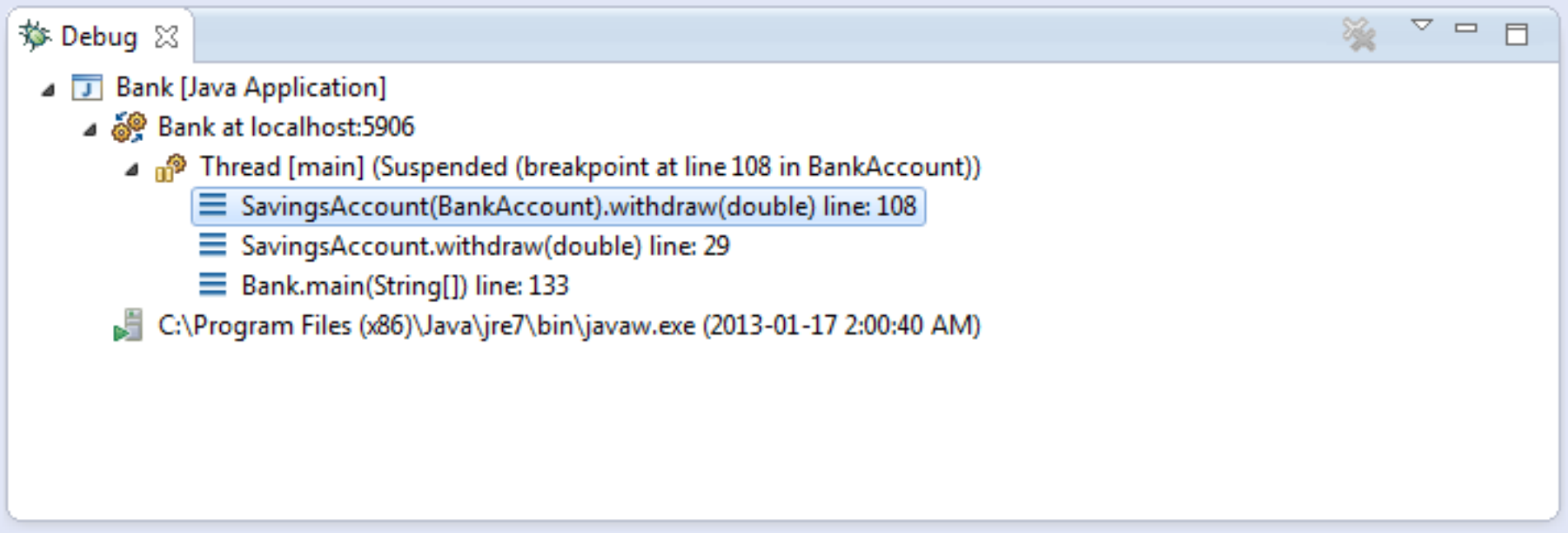
Eclipse Debug Features: Different Views

- Execution View



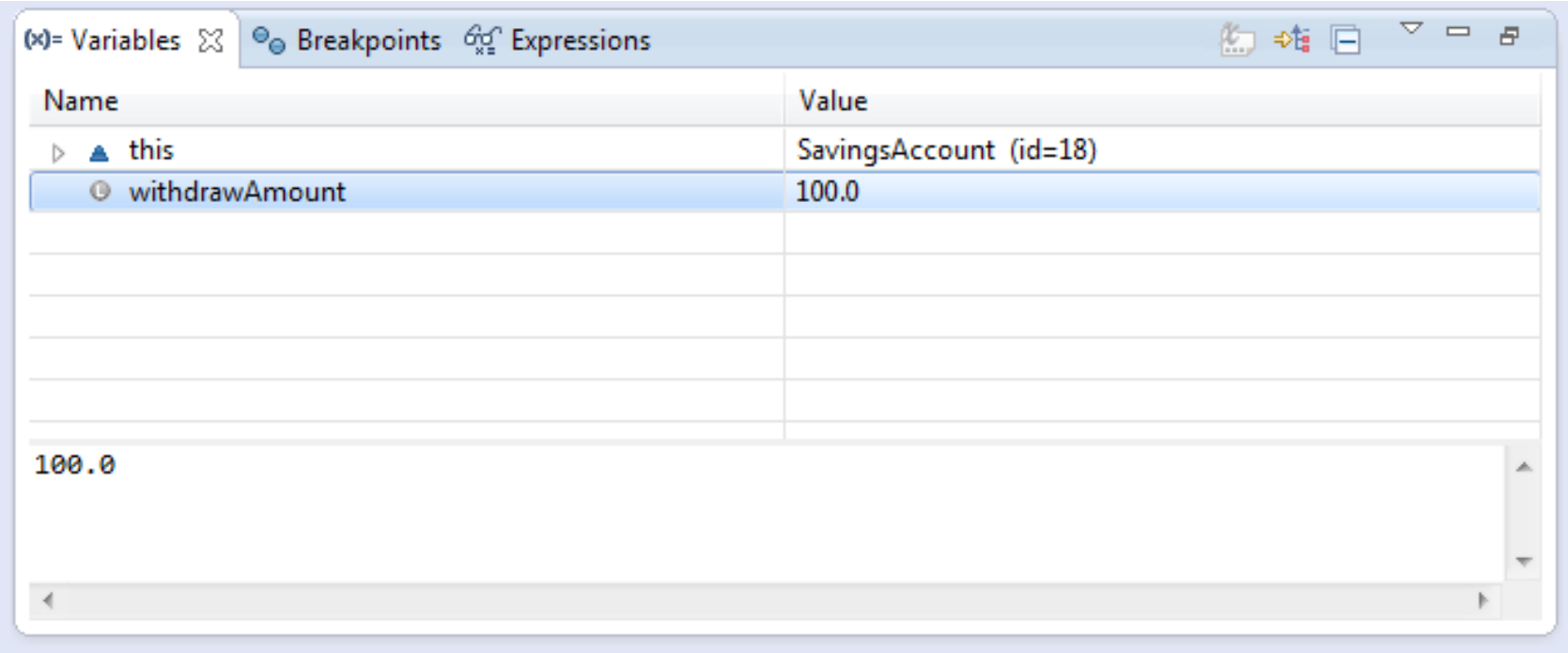
Eclipse Debug Features: Different Views cont'd.

- Stack View



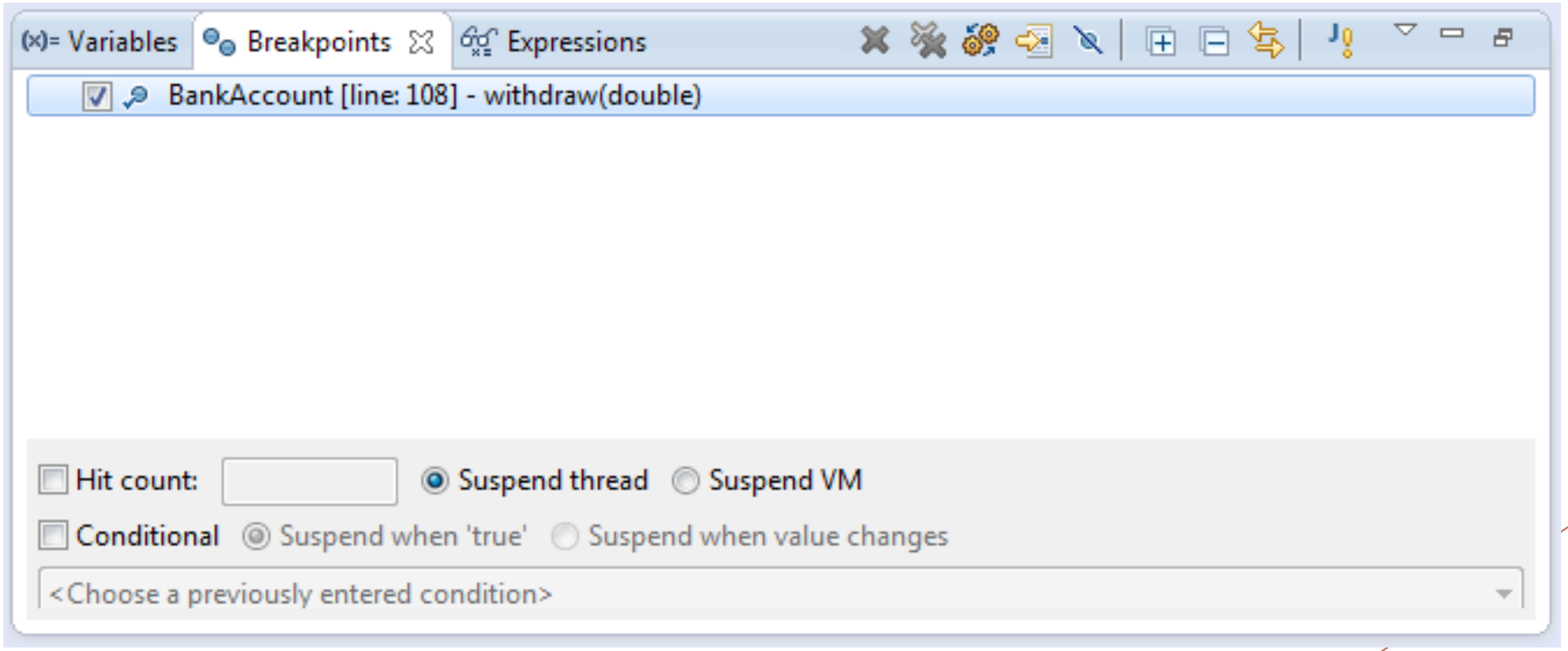
Eclipse Debug Features: Different Views cont'd.

- The Variables View

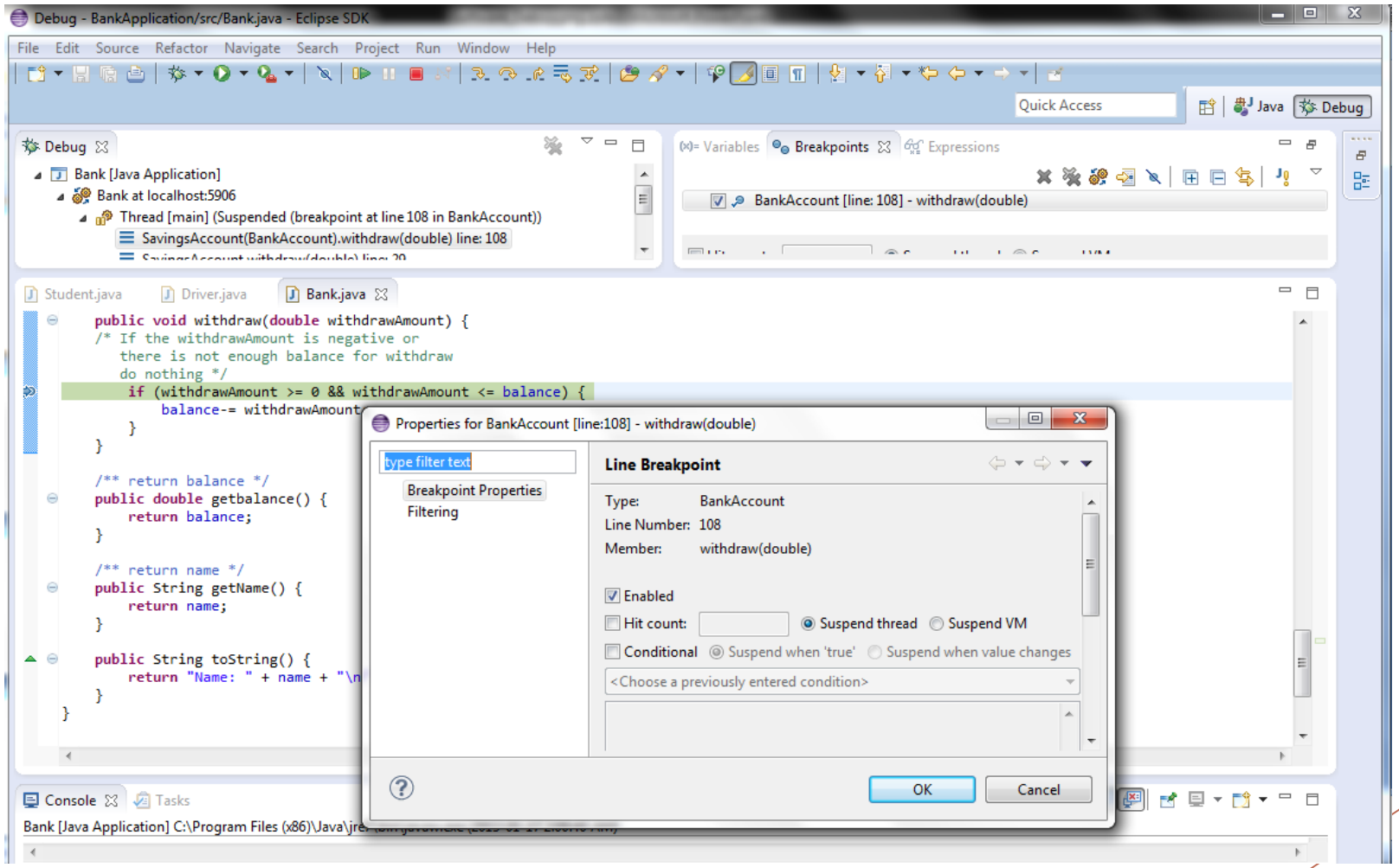


Eclipse Debug Features: Different Views cont'd.

- The Breakpoints View



Eclipse Debug Features:
Line breakpoint properties

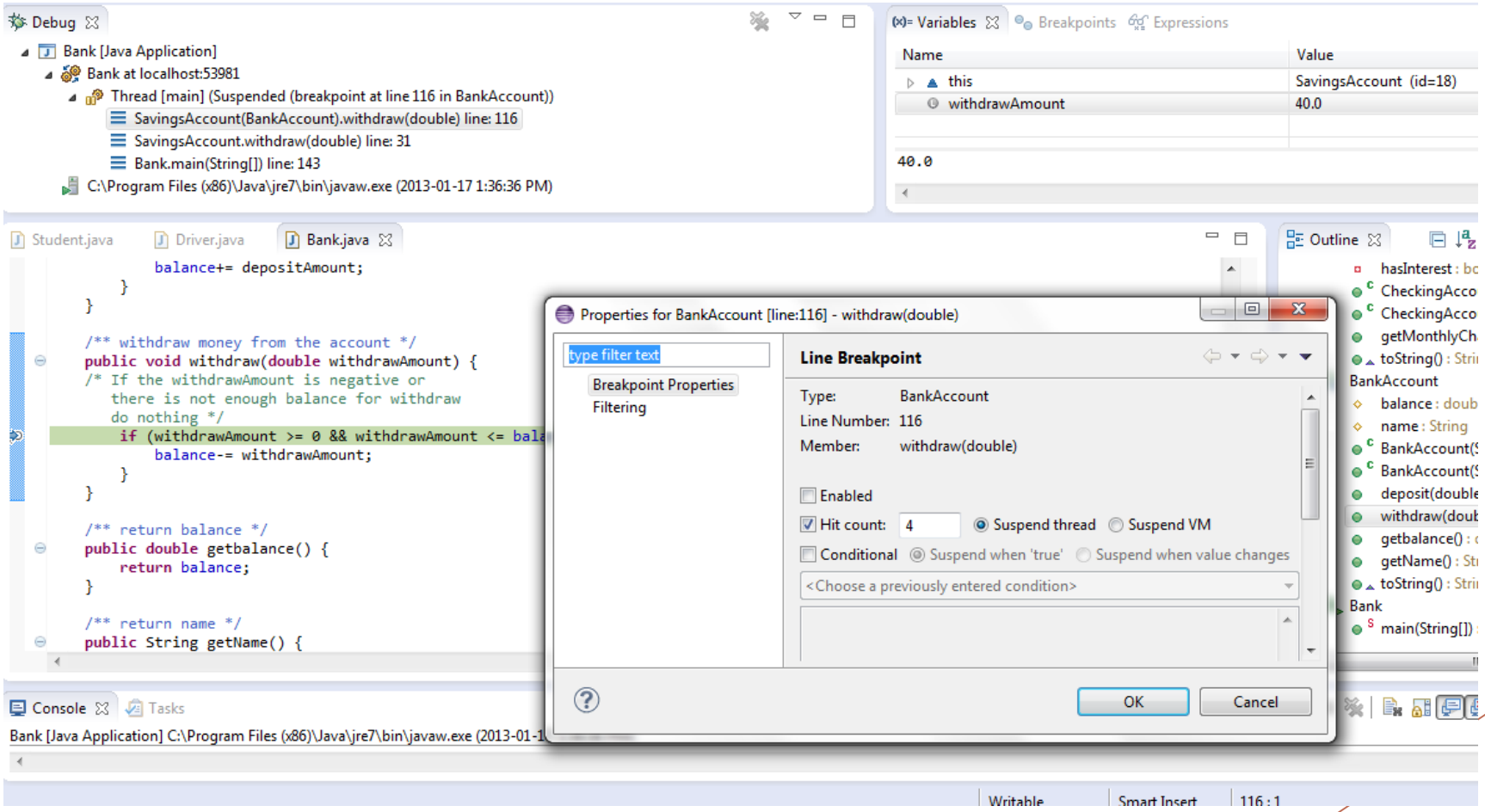


Eclipse Debug Features:
Line breakpoint properties cont'd.

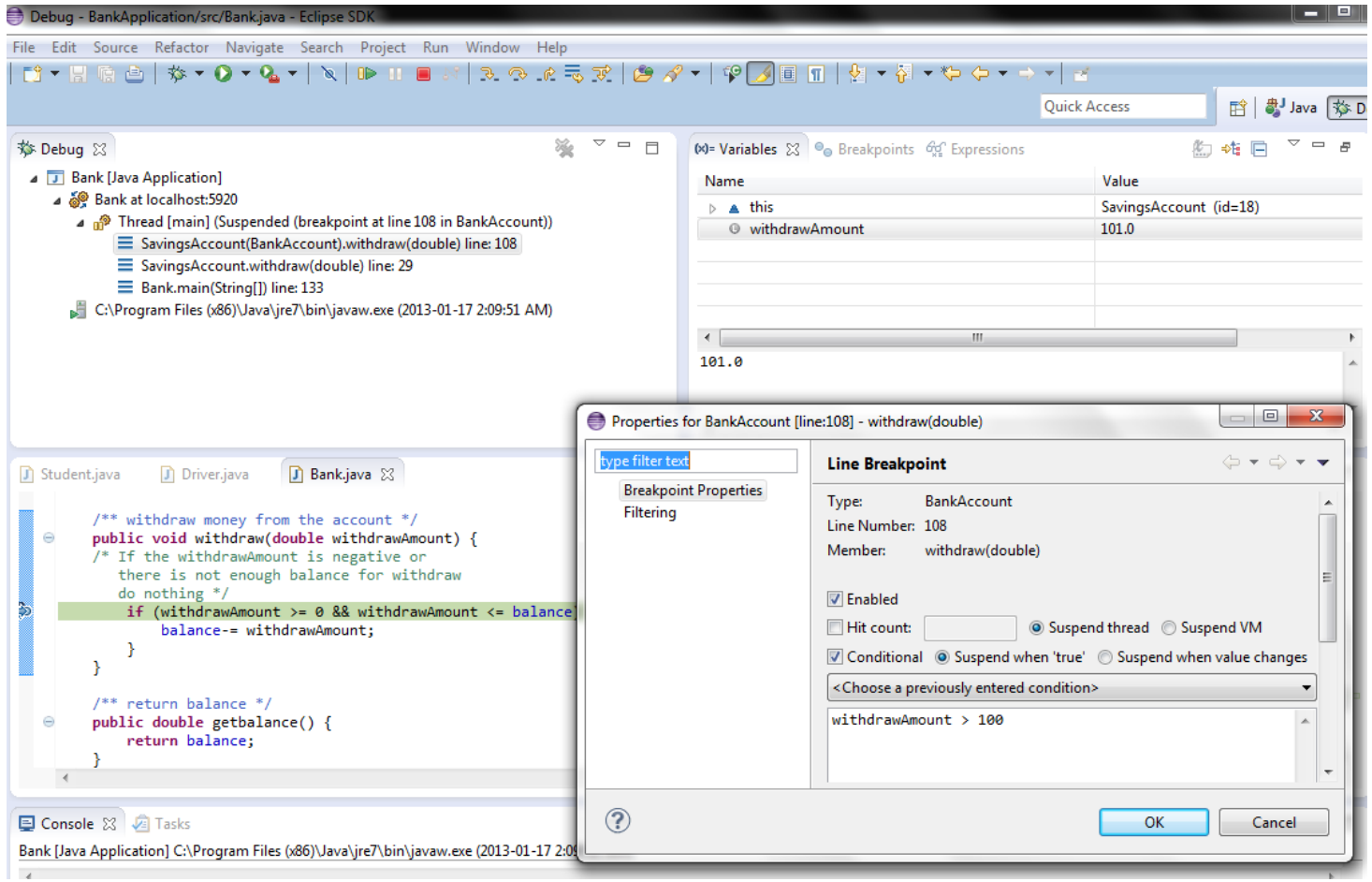
- Set 'hit count' break points
  - The program suspends at the particular break point when it reaches the 'hit count' limit.

- Set 'conditional' break points
  - The program suspends at the particular break point when the given condition is true.

Eclipse Debug Features:
Line breakpoint properties cont'd.

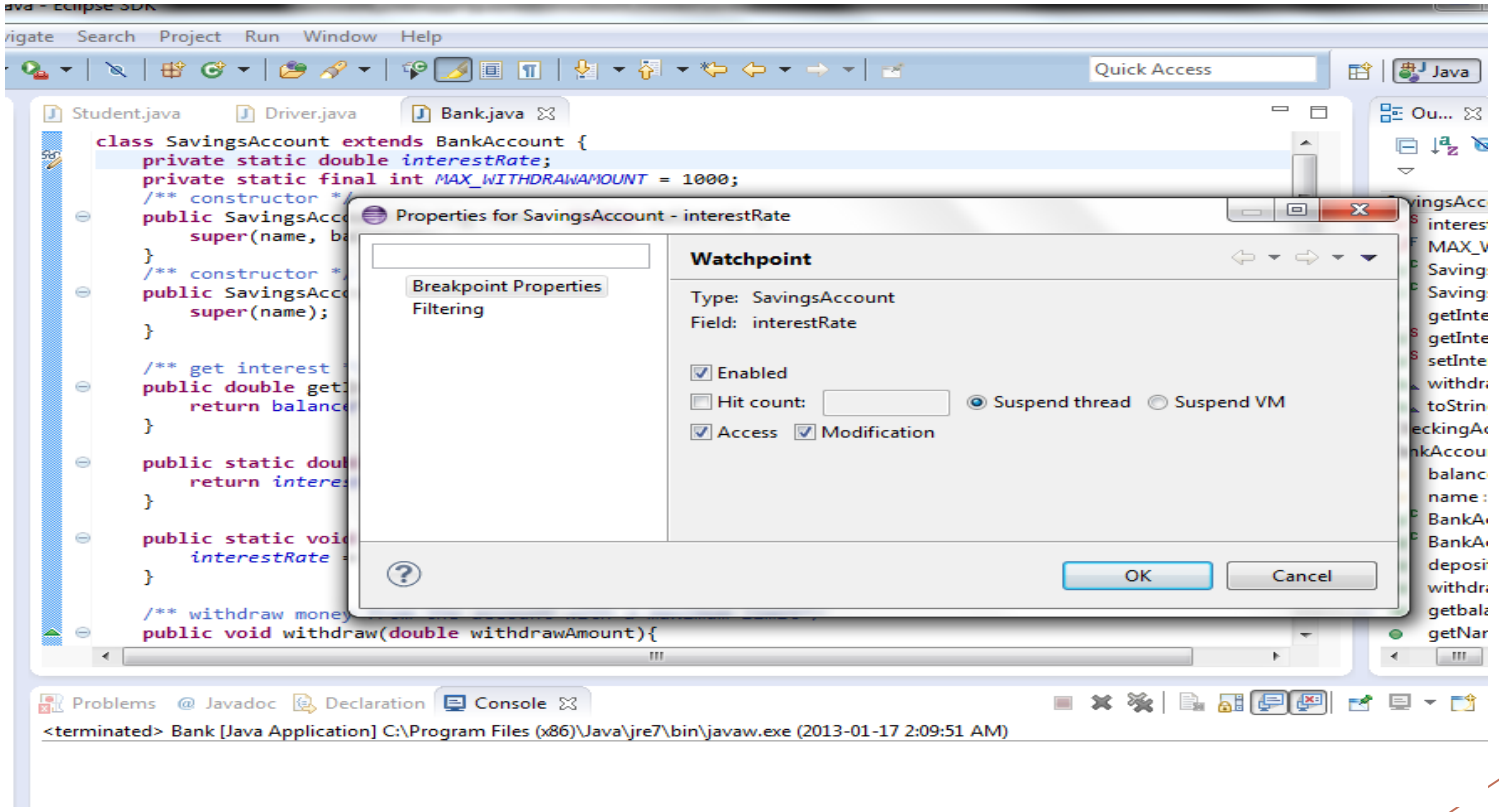


Eclipse Debug Features: Conditional line breakpoints cont'd.



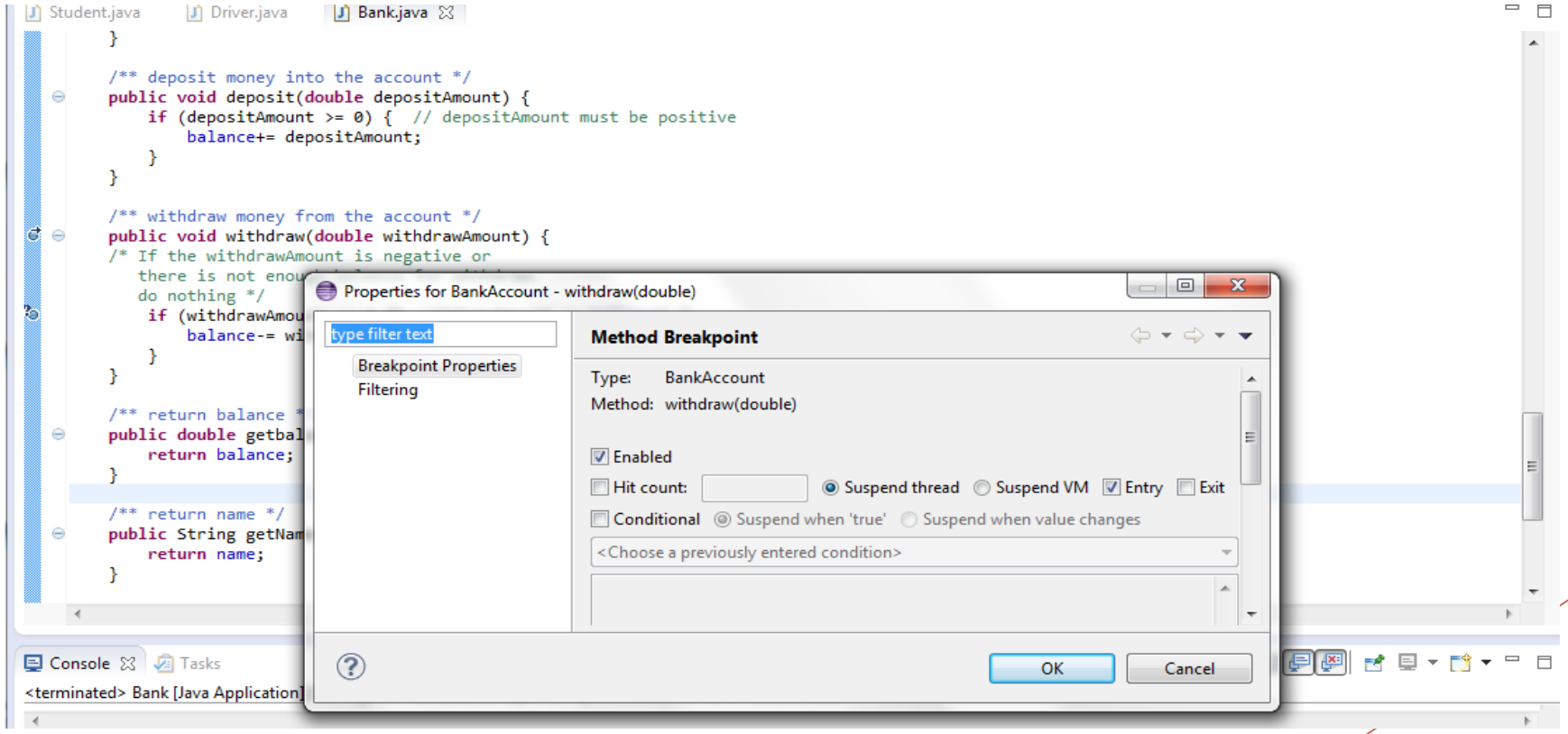
Eclipse Debug Features: watchpoints

- A watchpoint is a break point which targeted on a particular field.

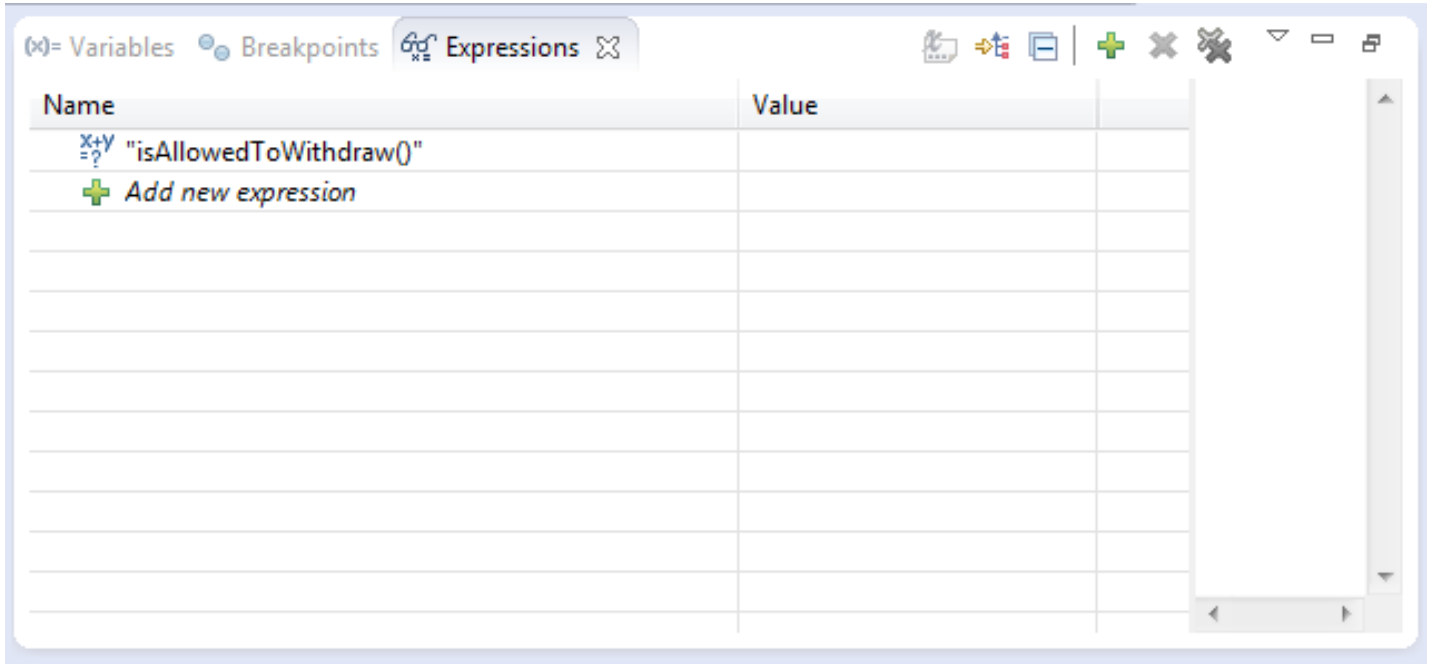


Eclipse Debug Features: Method breakpoints

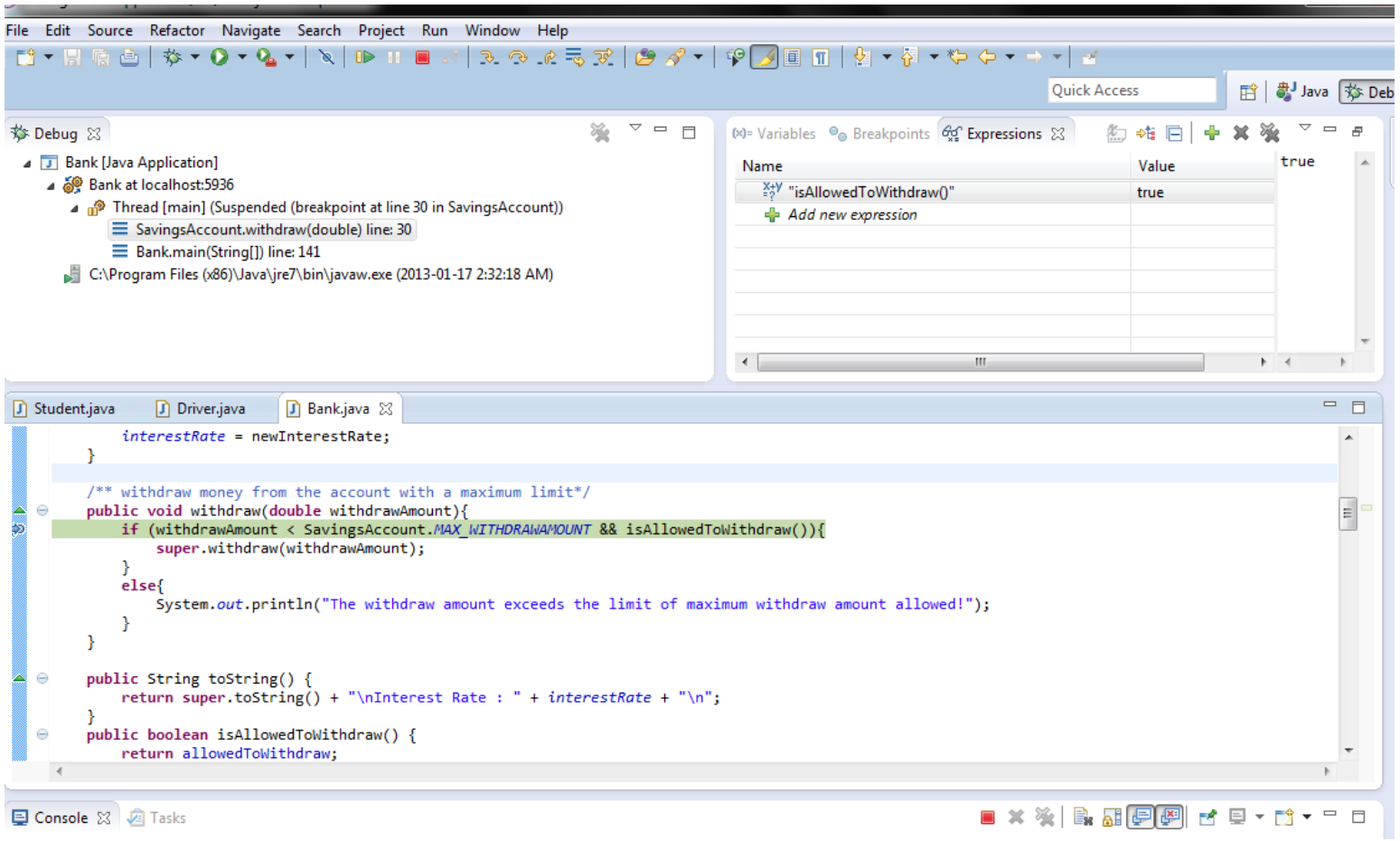
- A method breakpoint is a break point which targeted on a particular method.



Eclipse Debug Features: Expression View & Watch Expressions

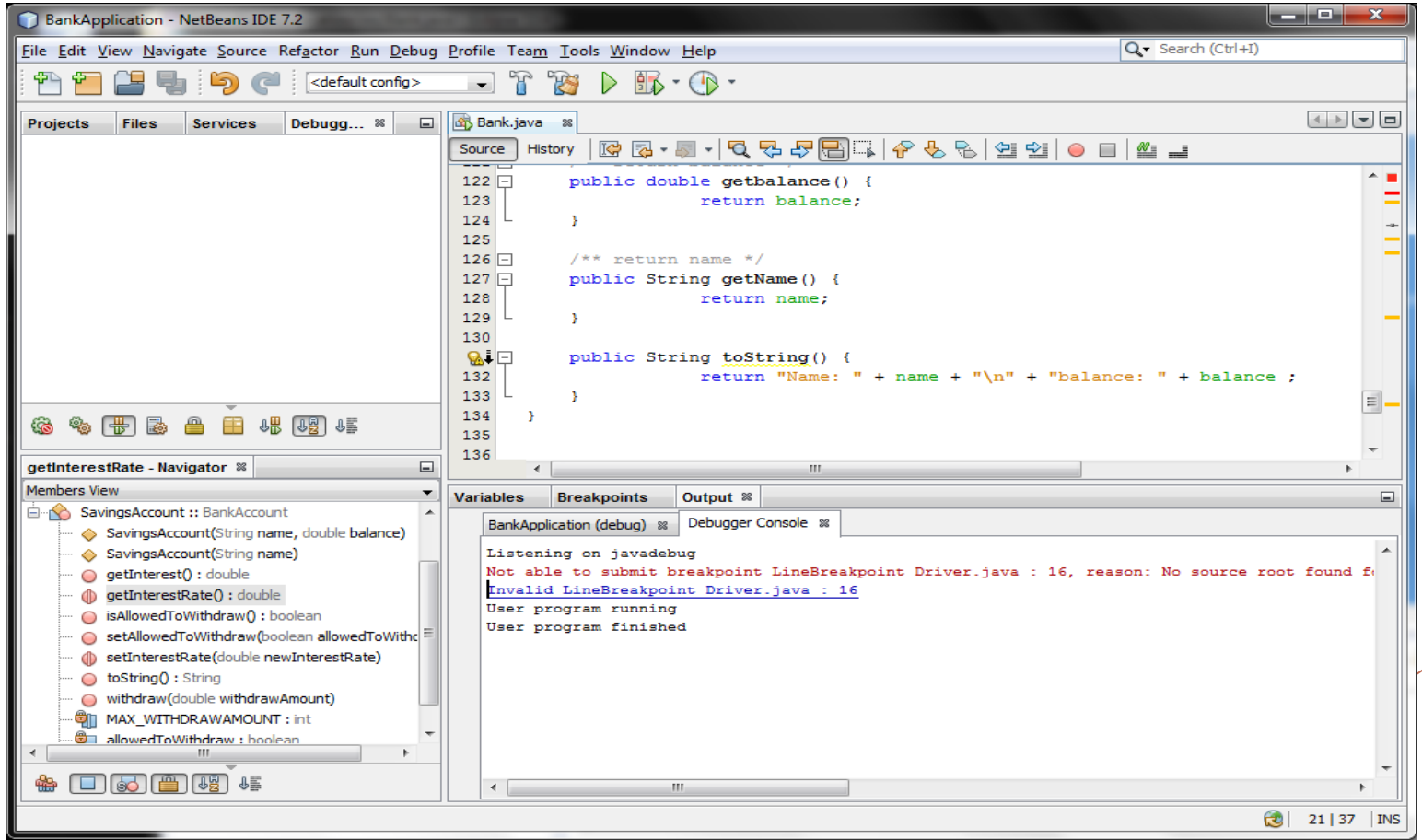


Eclipse Debug Features: Expression View & Watch Expressions cont'd.



Debug using NetBeans

- Basic features are more or less the same as in Eclipse.

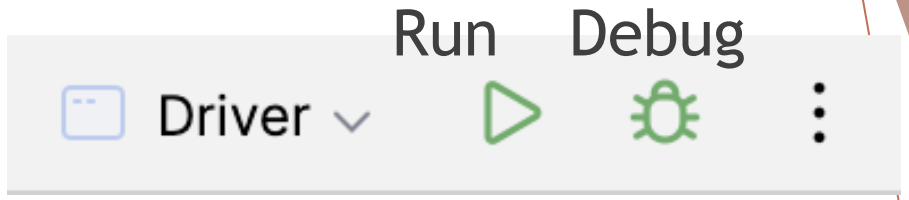


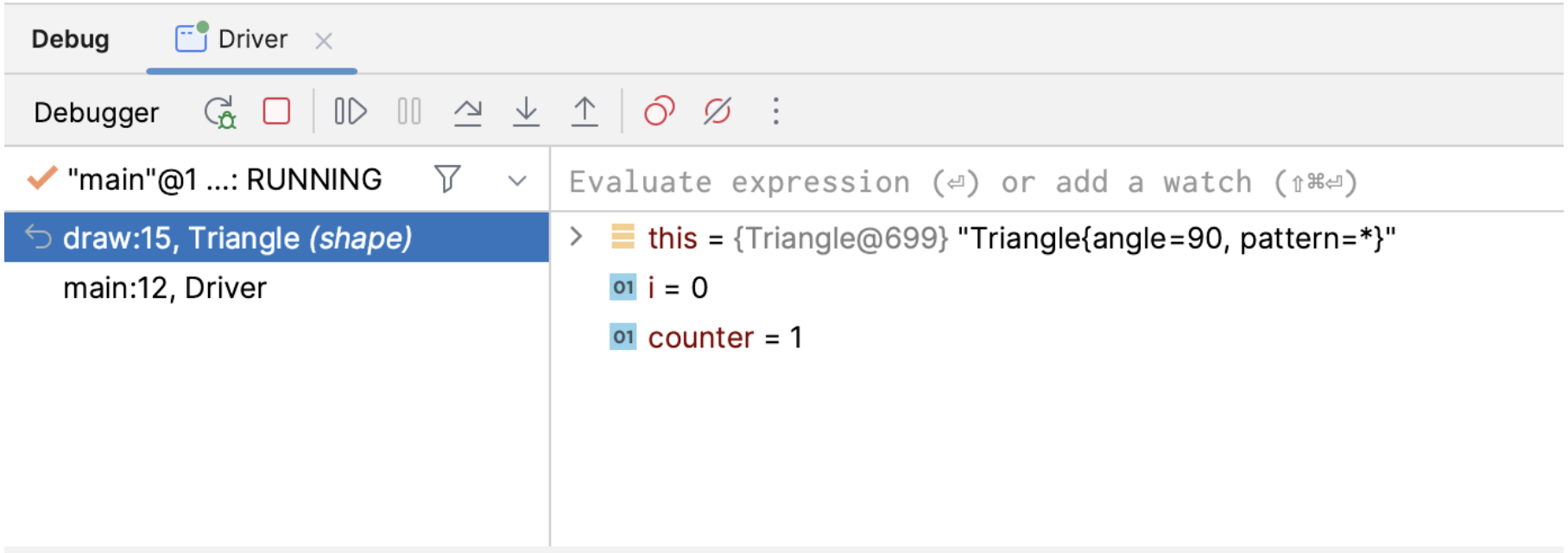
Debug using NetBeans cont'd.

- NetBeans also supports:
  - Line breakpoints.
  - Watchpoints
  - Method breakpoints

- Provide different views:
  - Code execution
  - Variables
  - Breakpoints

IntelliJ debug panel





Moral of the story...

- Try to avoid writing buggy code!
  - Design your program before putting it in to code.
  - Modularize your code.
  - Test continuously what has implemented so far.

Yet ... In reality, bug free code is a hard guarantee!

References

- http://en.wikipedia.org/wiki/Debugging
- http://www.vogella.com/articles/EclipseDebugging/article.html#debugging_overview
- http://en.wikipedia.org/wiki/List_of_software_bugs