Name: Suryaprabha K

Domain: BFS

Date:05-APR-2022

Debugging:

* Usually, while writing a piece of code, we use print statements to understand what is going on in the code.
* This method is suitable for smaller and lesser complex codes, it is not the best way to understand where your code is getting stuck.

What is Debugging:

* Debugging is a technique that is used to see the code execute line by line.
* While debugging, we can see the execution line by line of the written code and can stop at any line of code.
* We can analyze the code, variables, values assigned to those variables where the code stopped execution.

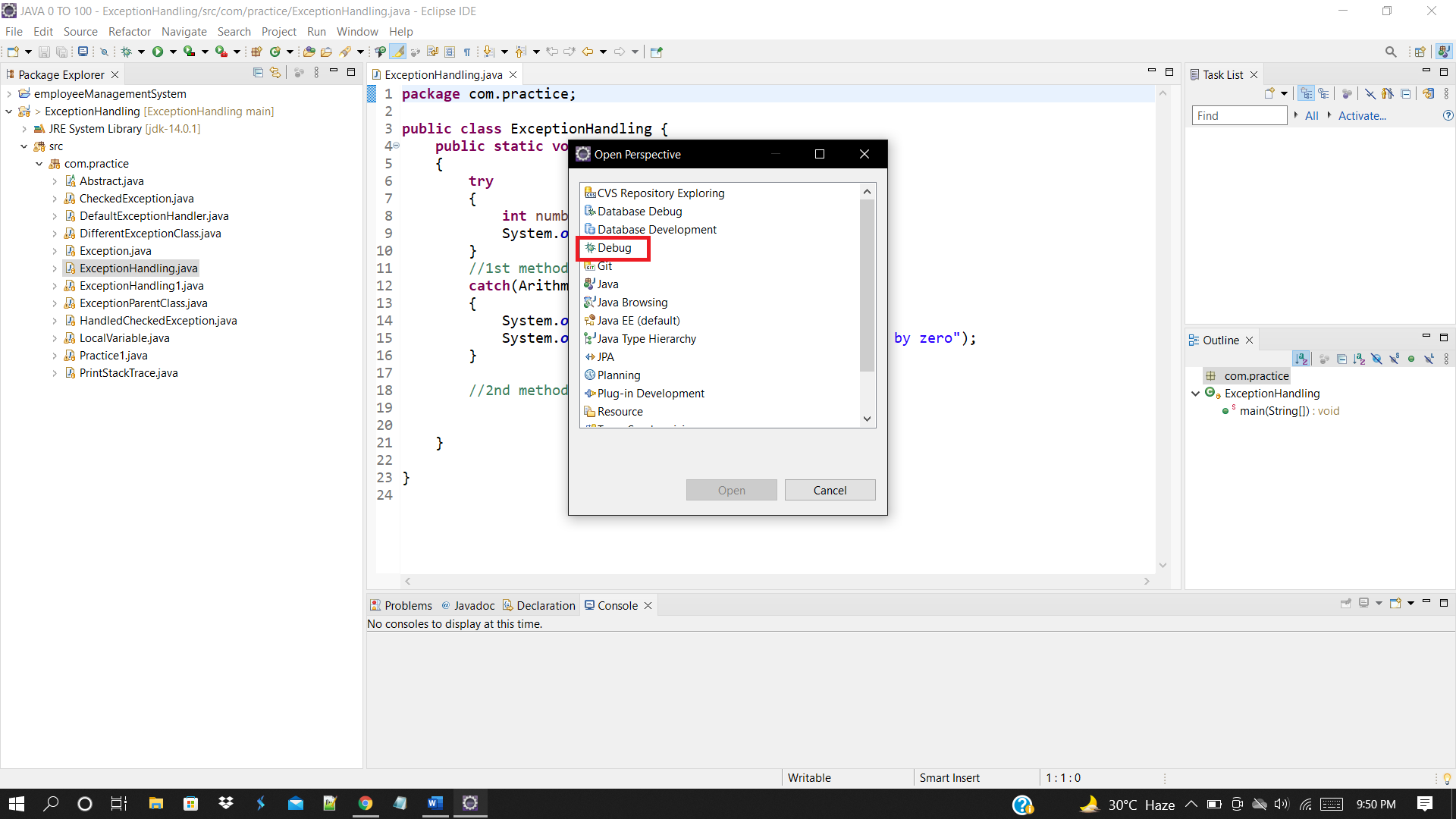
Purpose Of Debugging:

* Usually, compiler errors are clearly visible while writing the code in the eclipse.
* But run time errors are visible only when we execute the code and while running, if any run time error occurs, the program fails to execute and terminates abruptly.
* To find the run time errors, Debugging is used.
* To remove bugs, errors in the program which occurs only during specific condition.

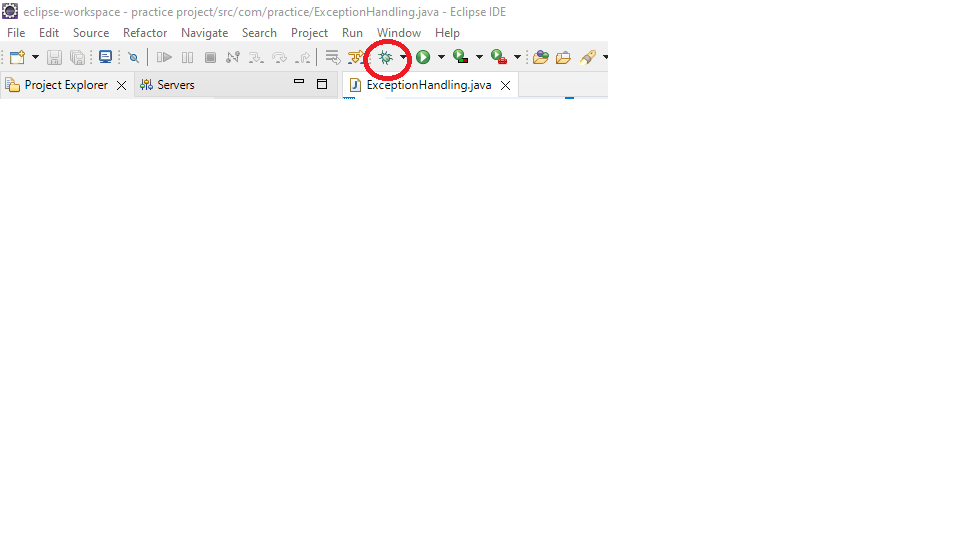
How to Launch Debugger in Eclipse?

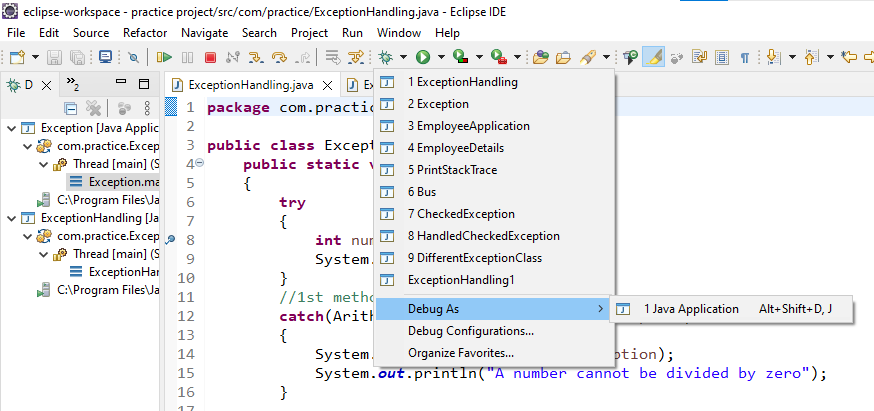
To launch Debugger,

Click on Open perspective->Choose Debug->Open

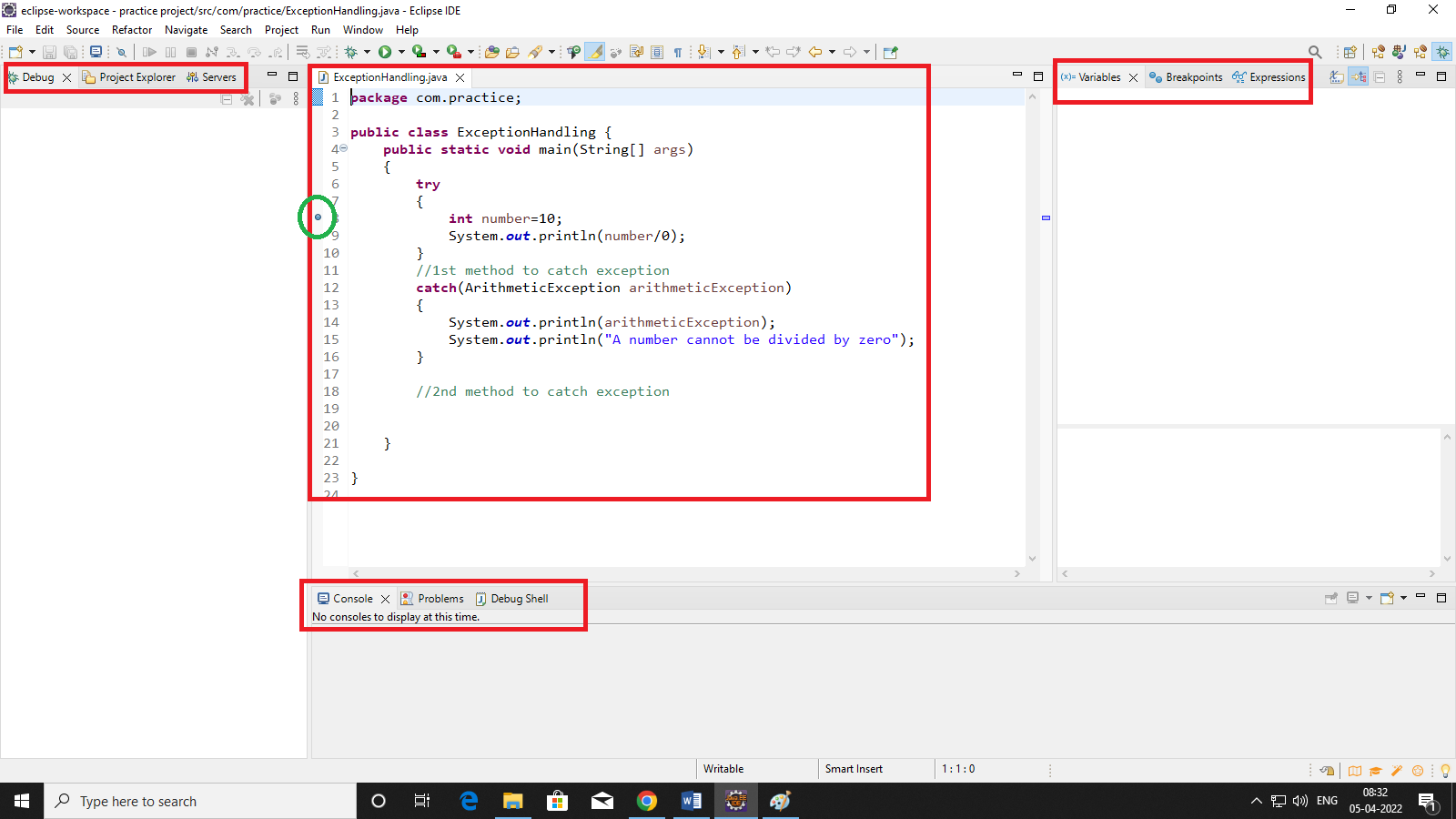


How to Debug?





Components in Debug Perspective:



* Debug Window
* Variables
* Console-Debug Shell
* Breakpoint
* Class

Debug Window:

* Debug window is present next to the project explorer.
* The class which is being debugged is displayed in the debug window.

Breakpoint:

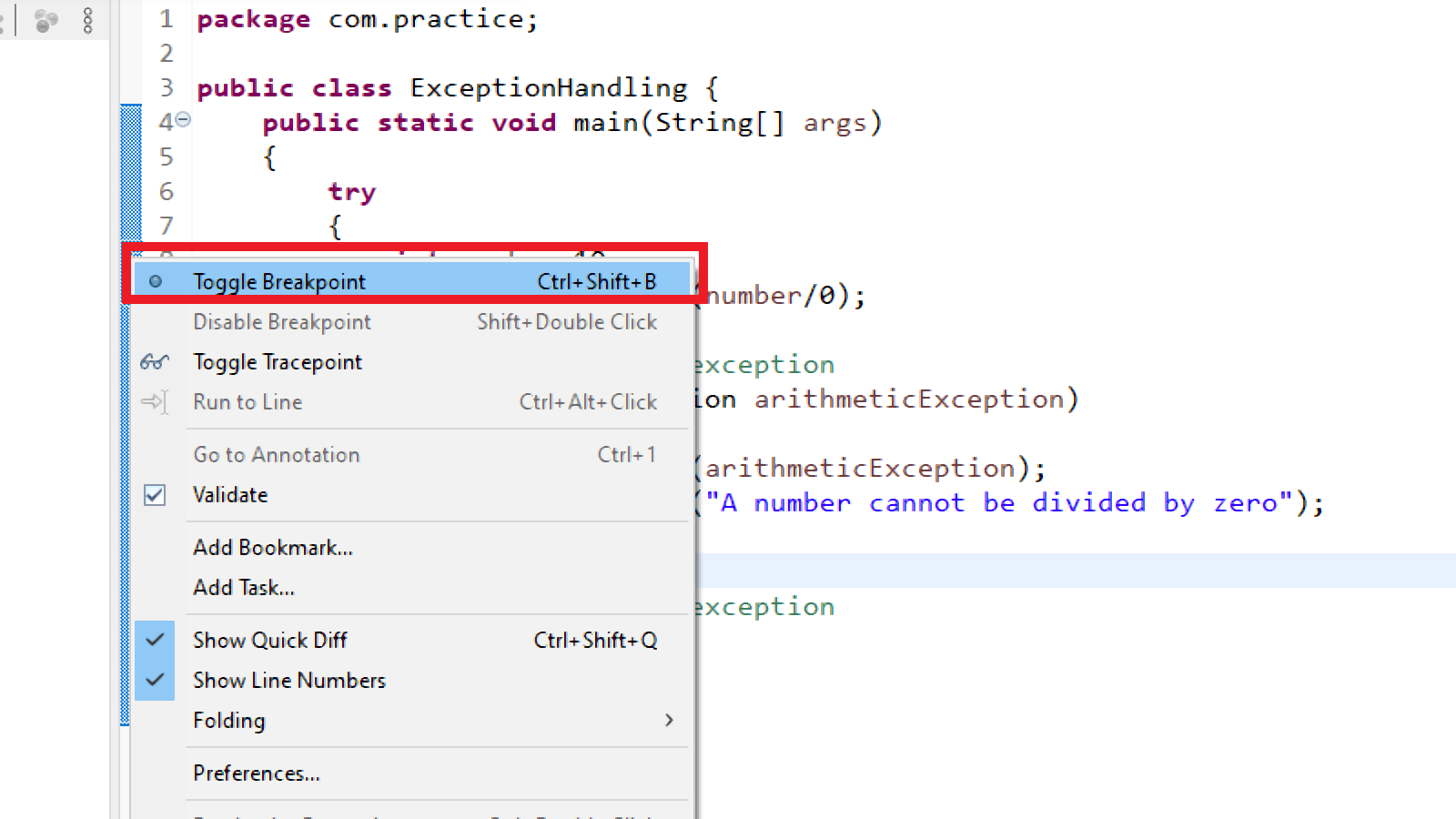
* A breakpoint is a point where the execution of the program is halted at that point when the program is in debug mode.

Why Breakpoint?

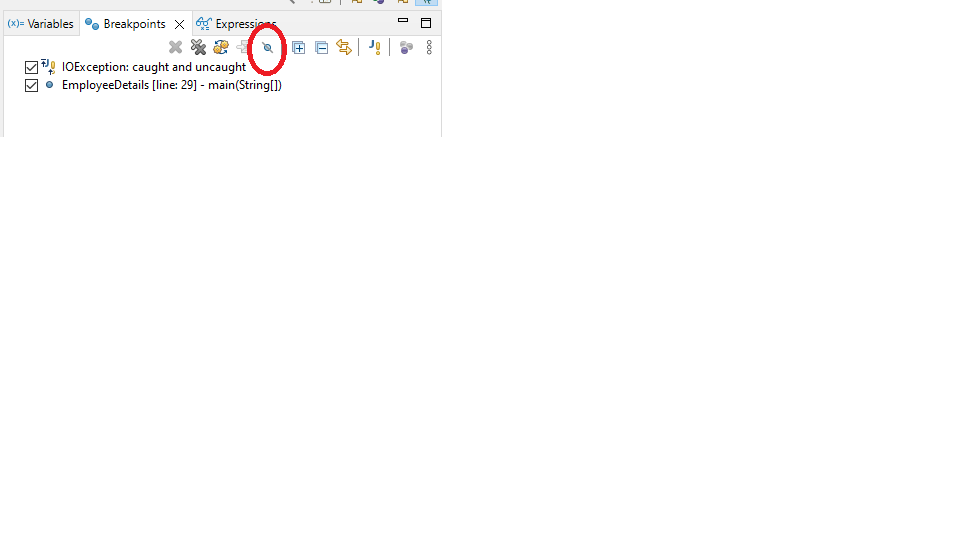
* To take manual control of a program at the time of execution, we need breakpoint.

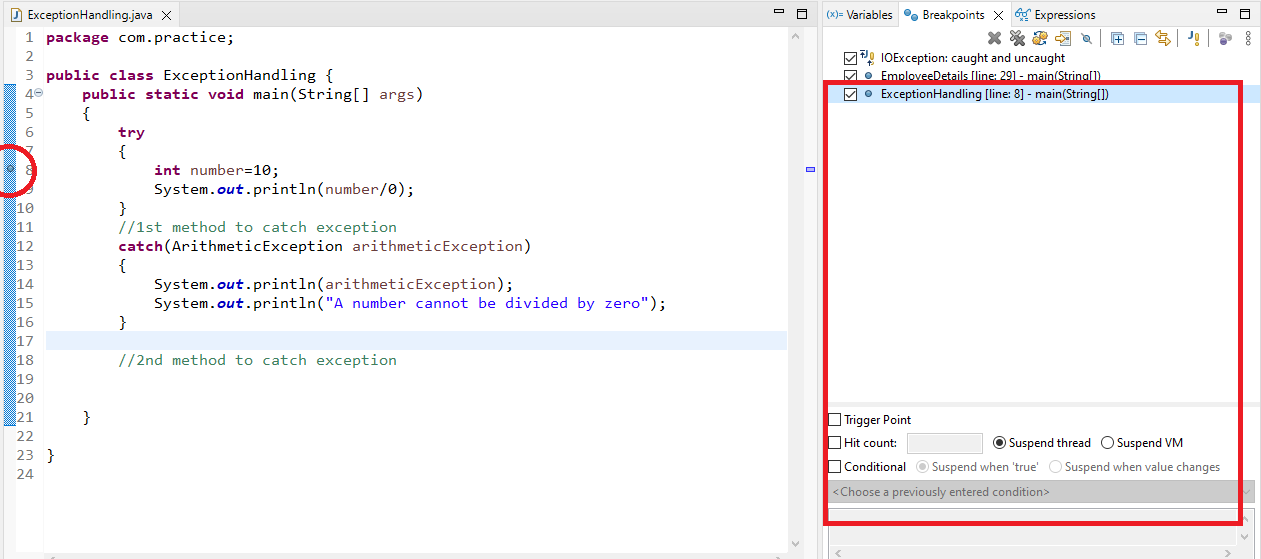
How to define a breakpoint?

* To define a breakpoint in a piece of code, right click on the left margin of the line in editor and choose Toggle breakpoint.
* Breakpoint is also defined by double clicking on the left margin of the line.



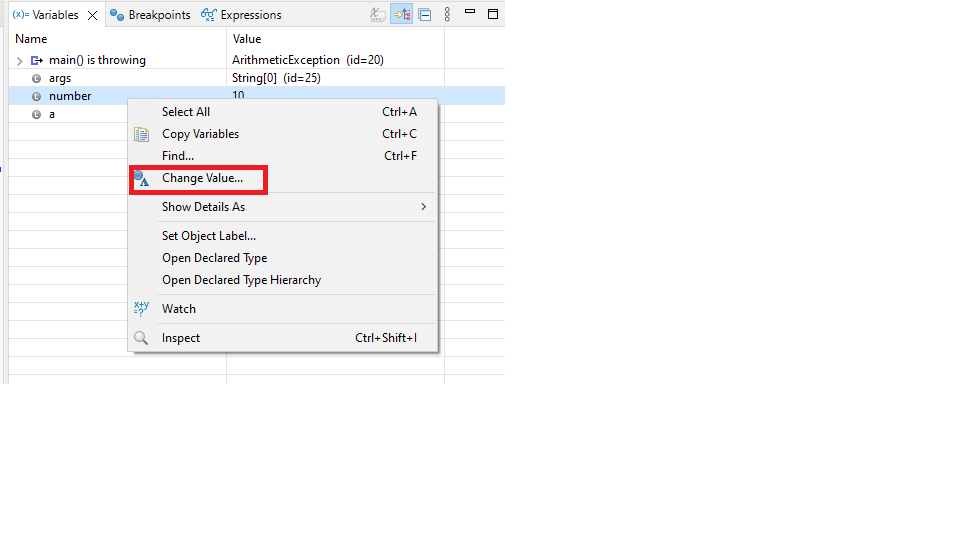
* When the program execution reaches the breakpoint, the program execution gets halted.
* Breakpoints view allows us to delete and deactivate the breakpoints.
* All breakpoints can be enabled/disabled using Skip All Breakpoints.

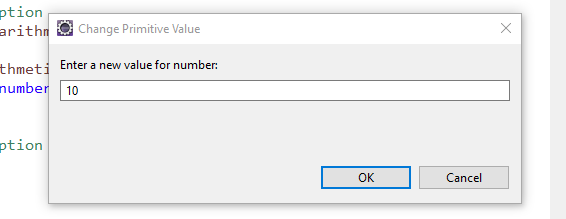




Variables:

* In this view, the variables can be viewed and how their state is changing during execution can be known.
* By right-clicking on the variables displayed here, we can change the value of those variables or view their data type, etc.



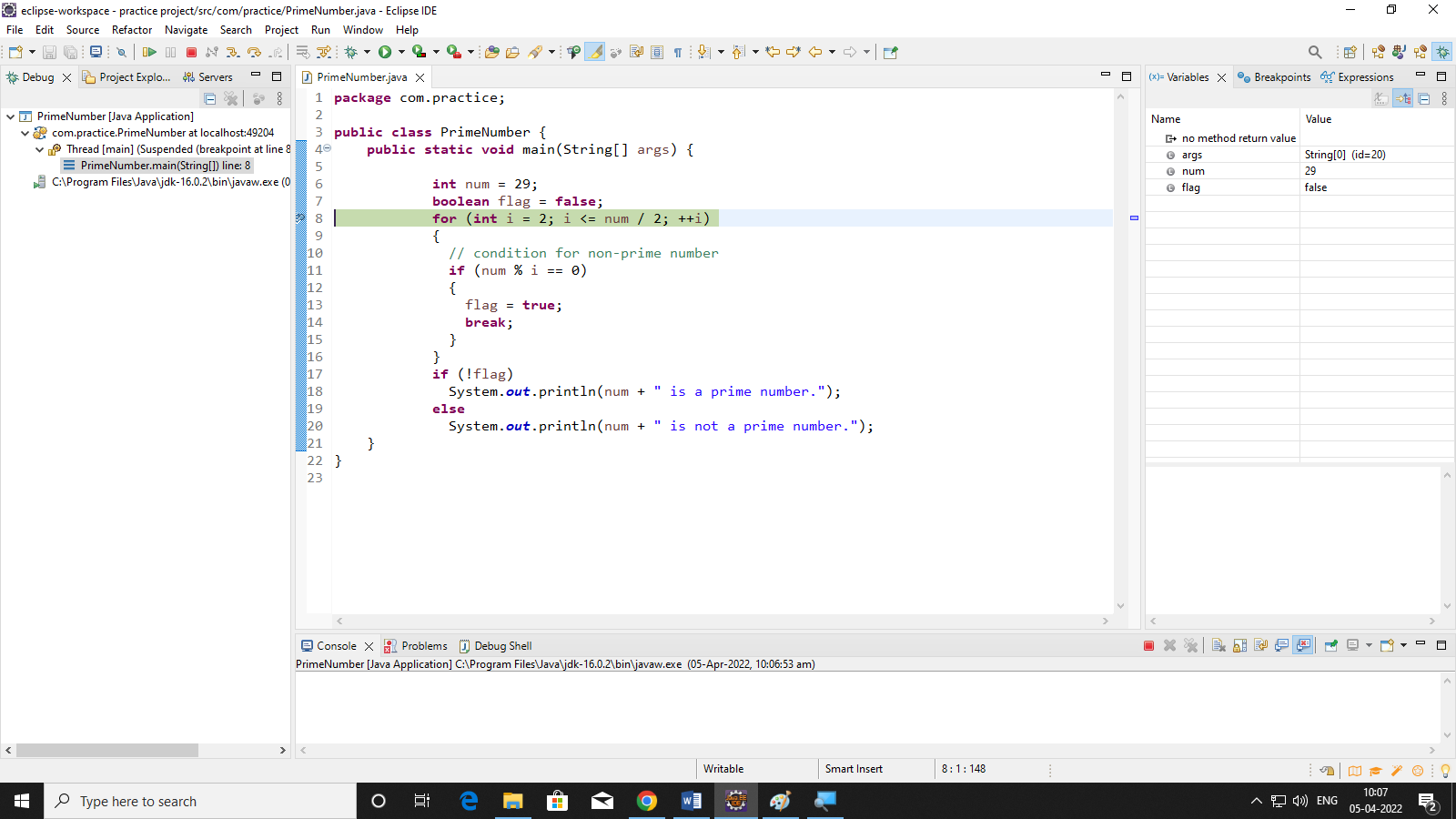


Console:

* This is where you can see the execution happening.

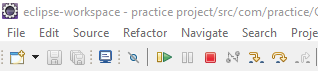
Process:

When we start debugging a particular piece of code, following window will appear.

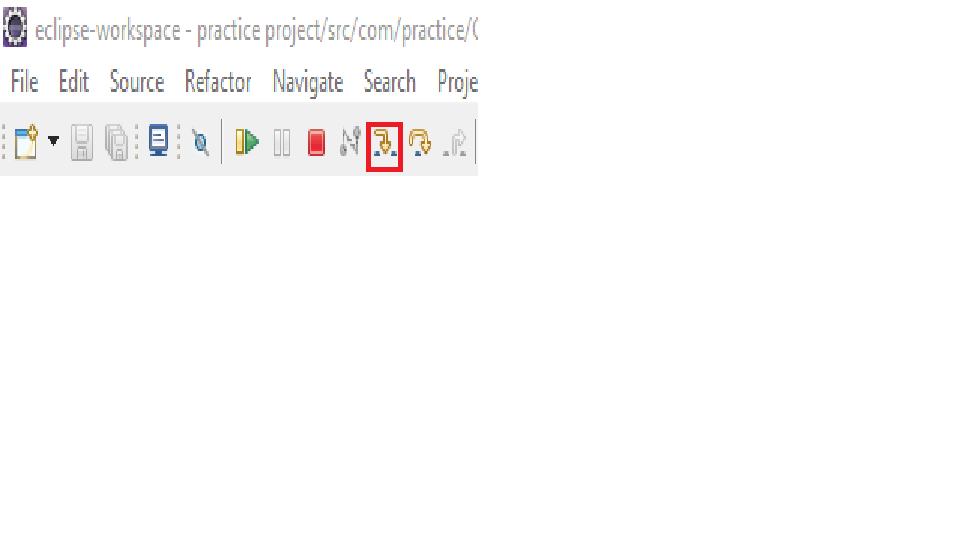


* The execution stops at the breakpoint here it is the main method.
* The lines of code above break point will be executed and after the breakpoint, the execution stopped.
* The variables that is declared above the breakpoint will be displayed in the variables view.

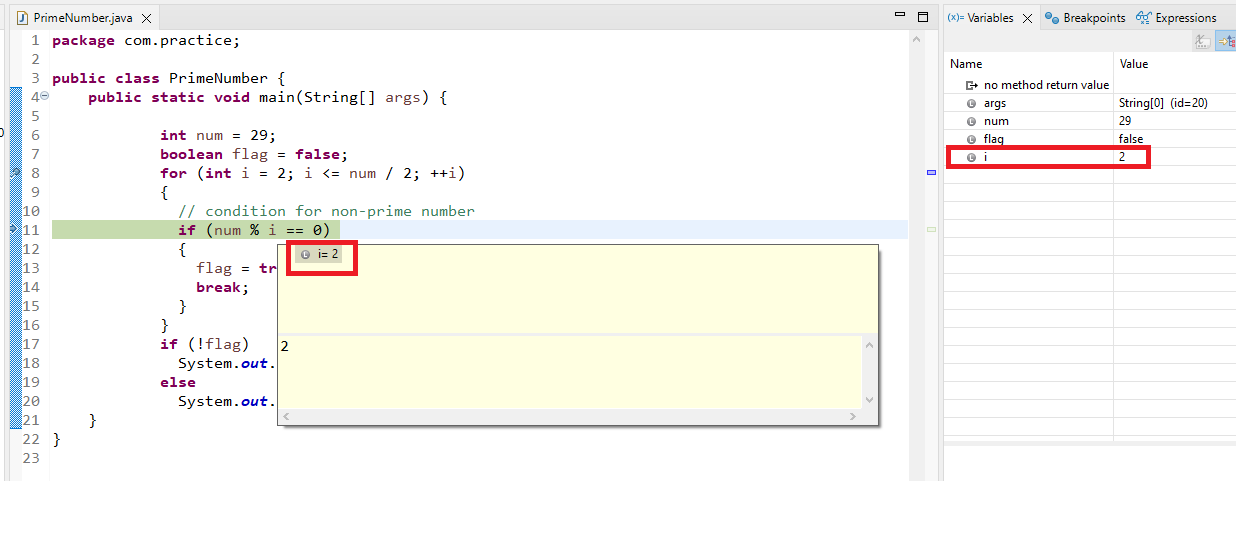
To manually enter the main method and analyze, certain steps are to be followed.



Step into or F5:



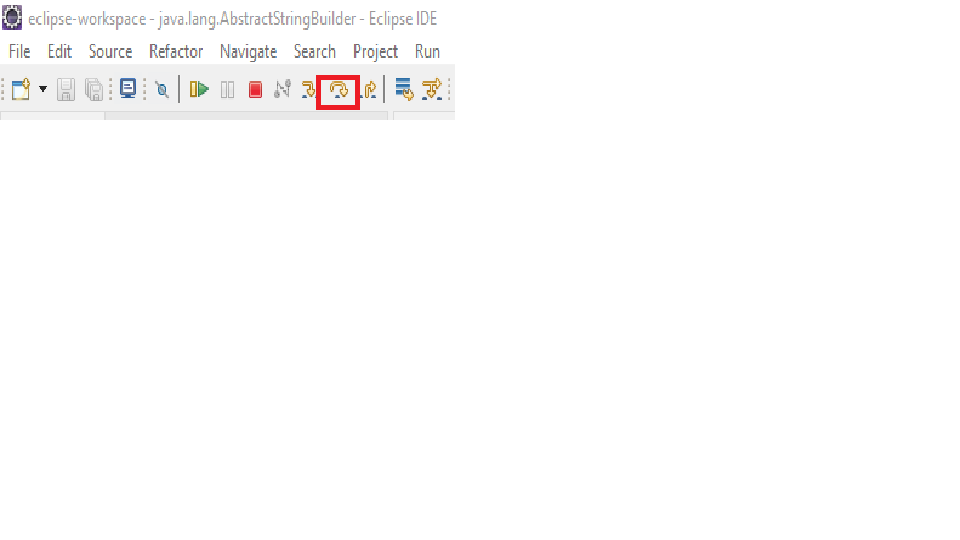
* Using Step into, we can execute the line of code where the breakpoint is at and move to the next line.



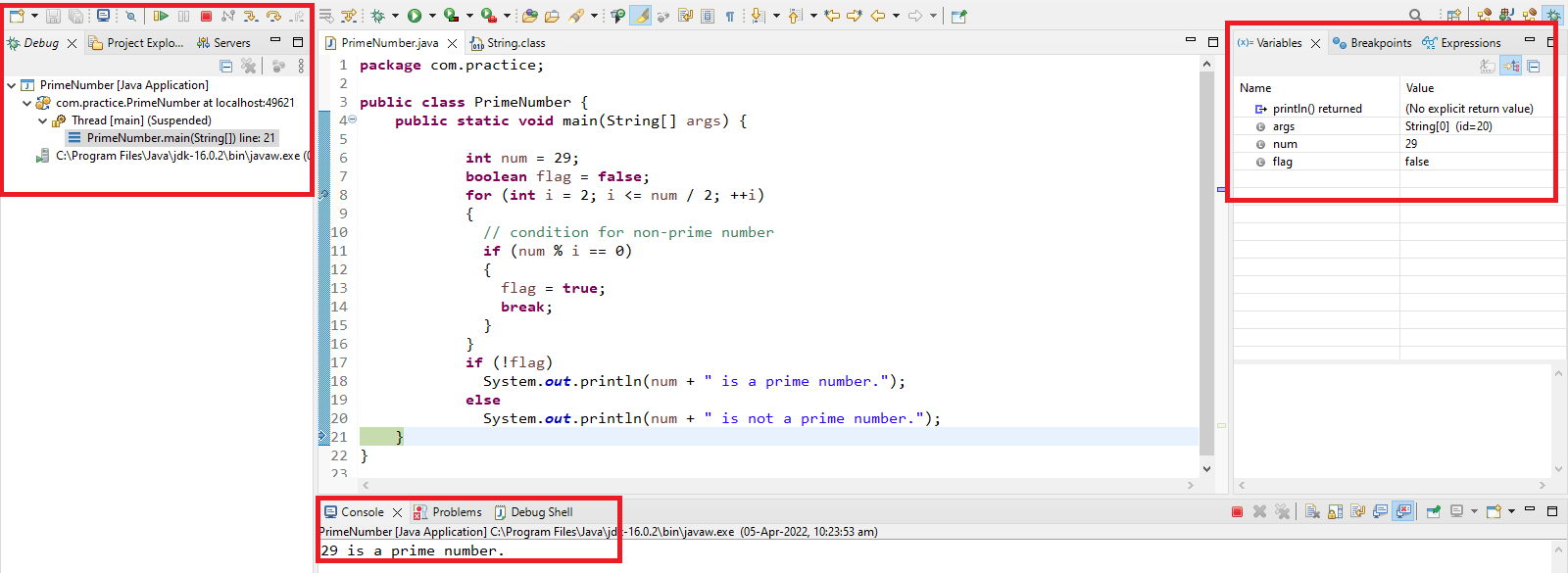
* To navigate through the for loop till the end, we need to step into again.
* Shortcut key to step into-F5

**Step over of F6:**

This option is also available as the icon shown below.

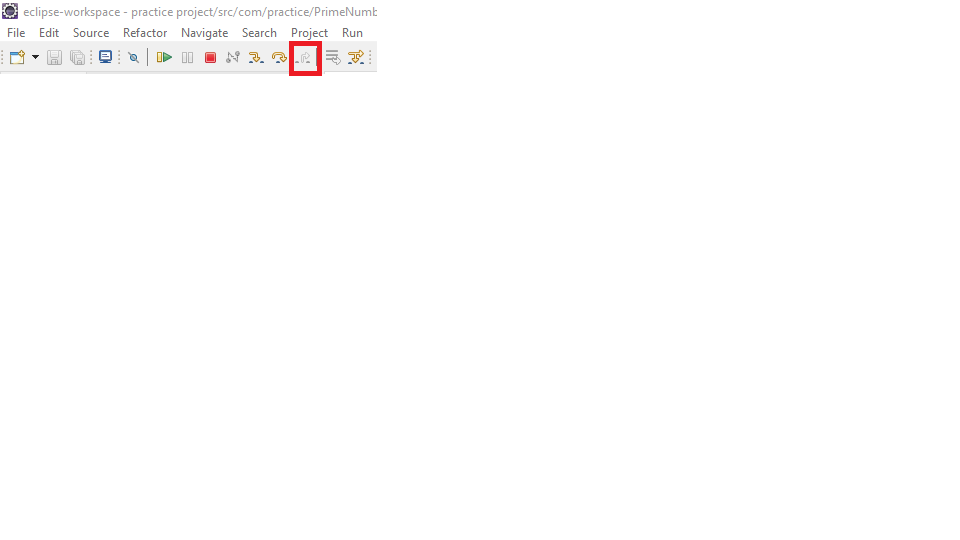


* Using this we can skip the debugger and execute the code.
* In this case, the code will execute normally till you keep hitting F6 and in the end, you will get the exception as you do while normally executing.



**Step out or Step return or F7:**

* This can be used when your code is in a method and is being called from another place.
* This key will finish the execution of the method being debugged and return to the code from where this method is being called.
* This can also be done by clicking on the below icon.



**Resume or F8:**

* This option will tell the debugger to continue executing the program until the next breakpoint is reached.

