Hyperiondev

TASK

Debugging

Visit our website

Introduction

WELCOME TO THE DEBUGGING TASK!

Debugging is essential to any Software Developer! In this task, you will learn to use your IDE to debug your code more effectively.

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact your mentor to get support on any aspect of your course.

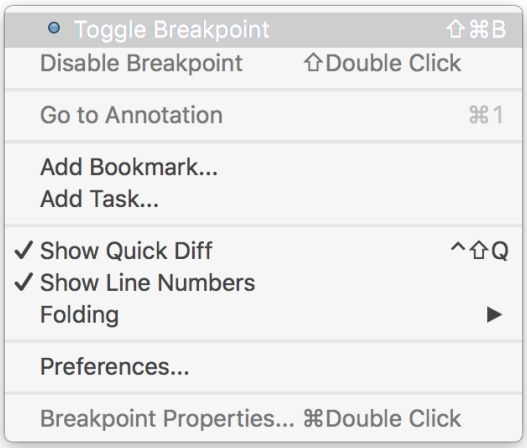
The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

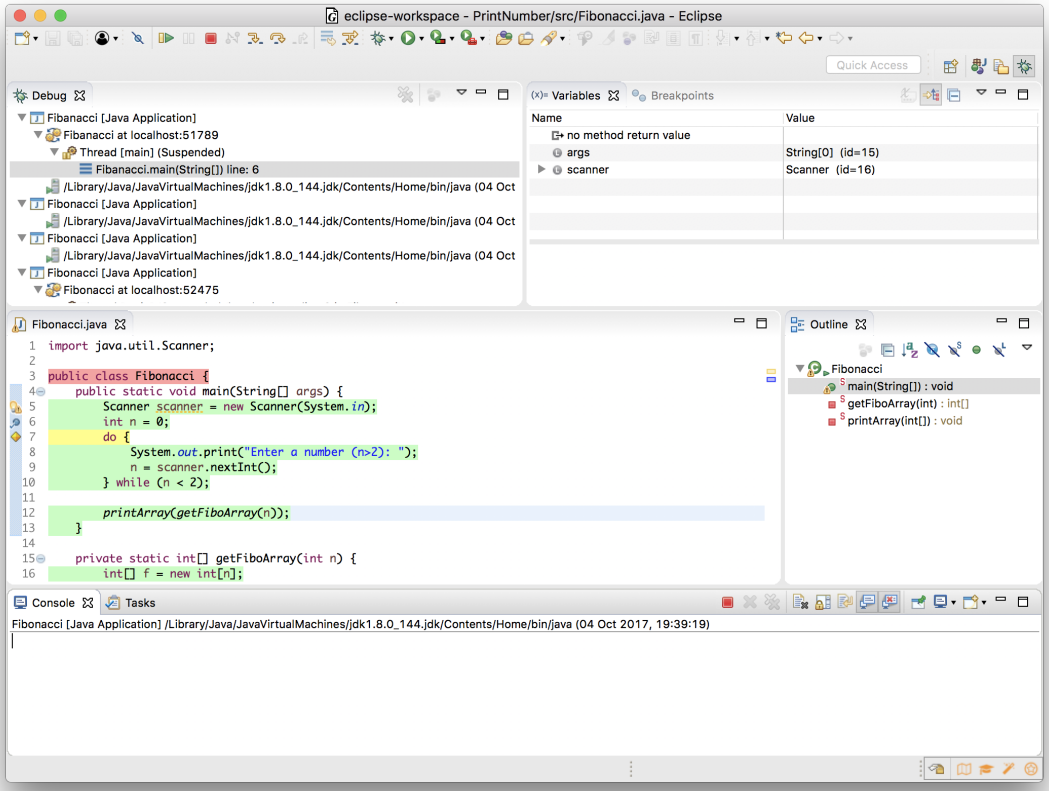
DEBUGGING WITH ECLIPSE

As you experienced with IDLE, Eclipse (and most other IDEs) have built-in features that support debugging. As you did with IDLE, you can also add breakpoints to your Java code. A breakpoint tells the debugger to suspend execution at a certain point in the code temporarily. You can do this to view information about a program's state.

To create a breakpoint, right-click on the left margin in the Java editor and select *Toggle Breakpoint* or double-click in the position you would like to create the breakpoint. A blue dot will appear next to the line of code where the execution will halt.



The Breakpoints view at the bottom of your screen allows you to delete and deactivate Breakpoints and modify their properties. When you run the application, you should see the debug perspective, which might look similar to the screenshot below. The debug perspective offers additional views that can be used to troubleshoot an application.



Let's take a closer look at the views in this perspective (for more information, have a look at **this link**):

- **Debug:** this view shows the live stack trace of methods. Here we are only in the main method of the program. If other programs were running, we would see them here.
- **Variables:** this view shows all the variables that have been declared and their values. We do not see *n* yet because our breakpoints stop before it is declared.
- **Breakpoints:** this view shows the location of all the breakpoints in your code.
- **Console:** this view shows the output of the program.

Eclipse has various buttons in the toolbar and key binding shortcuts to control program execution. These help developers debug their programs. You can learn more about these stepping commands **here**.

Compulsory Task 1

- Follow the instructions above on how to debug a program using Eclipse.

- Create a simple calculator application that asks a user to enter two numbers and the operation (e.g. +, -, x, etc.) that they'd like to perform on the numbers. Display the answer to the equation. Every equation entered by the user should be written to a text file. Use defensive programming to write this program in a manner that is robust and handles unexpected events and user inputs.

Rate us
Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.

References:

Loudoun County Public Schools. (n.d.). *Stepping through the main method*. Retrieved from https://www.lcps.org/cms/lib/VA01000195/Centricity/Domain/5120/Debugger.docx

Oracle. (2019). *The Catch or Specify Requirement*. Retrieved from The Java Tutorials: https://docs.oracle.com/javase/tutorial/essential/exceptions/catchOrDeclare.html

Oracle. (2019a). What Is an Exception? Retrieved from The Java Tutorials: https://docs.oracle.com/javase/tutorial/essential/exceptions/definition.html