

Intro to Java Week 5 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

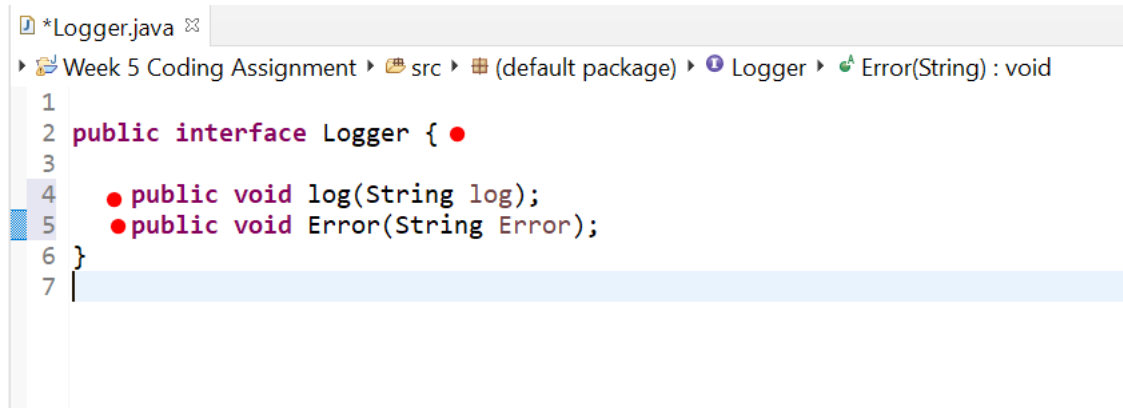
1. Create an interface named `Logger`.
2. Add two void methods to the `Logger` interface, each should take a `String` as an argument
 - a. `Log`
 - b. `Error`
3. Create two classes that implement the `Logger` interface
 - a. `AsteriskLogger`
 - b. `SpacedLogger`
4. The `log` method on the `AsteriskLogger` should print out the `String` it receives between 3 asterisks on either side of the `String` (e.g. if the `String` passed in is "Hello", then it should print `***Hello***` to the console.
5. The `error` method on the `AsteriskLogger` should print the `String` it receives inside a box of asterisks, with the `String` preceded by the word "ERROR:". For example, if "Hello" is the argument, the following should be printed:

Error: Hello

6. The SpacedLogger should add spaces between each character of the String argument passed into its methods.
7. If the log method received “Hello” as an argument, it should print H e l l o
8. The error method should do the same, but with “ERROR:” preceding the spaced out input (i.e. ERROR: H e l l o)
9. Create a class named App that has a main method.
10. In this class instantiate an instance of each of your logger classes that implement the Logger interface.
11. Test both methods on both instances, passing in Strings of your choice.

Screenshots of Code and Code Running in Application:

Steps 1 & 2 (Logger):



```
*Logger.java
Week 5 Coding Assignment > src > (default package) > Logger > Error(String) : void
1
2 public interface Logger {
3
4     public void log(String log);
5     public void Error(String Error);
6 }
7 |
```

Steps 3a, 4, & 5 (AsteriskLogger):

```
2 public class AsteriskLogger implements Logger {
3 // The log method on the AsteriskLogger should print out the String i
4 // (e.g. if the String passed in is "Hello", then it should print
5 // 5. The error method on the AsteriskLogger should print the Stri
6 // For example, if "Hello" is the argument, the following should
7 //*****
8 /**Error: Hello**
9 //*****
10
11 @Override
12 public void log(String log) {
13     System.out.println("***" + log + "***");
14 }
15
16 @Override
17 public void Error(String Error) {
18     System.out.println("*****");
19     System.out.println("***ERROR: " + Error + "***");
20     System.out.println("*****");
21 }
22
23 }
```

Problems @ Javadoc Console

<terminated> App (1) [Java Application] C:\Users\w lind\p2\pool\plugins\org.eclipse.justj.openjdk.hot

```
***Hello***
*****
***ERROR: Hello***
*****
H e l l o
ERROR: H e l l o
```

Steps 3b, 6, 7, & 8 (SpacedLogger):

```
Logger.java  AsteriskLogger.java  *SpacedLogger.java  App.java  *wk5Me
Week 5 Coding Assignment > src > (default package) > SpacedLogger > Error(Str
7  @Override
8  public void log(String log){
9      StringBuilder result = new StringBuilder(); // iterates o
10     for (int i = 0; i < log.length(); i++) {
11         if (i > 0) {
12             result.append(" ");
13         }
14         result.append(log.charAt(i));
15     }
16     System.out.println(result.toString());
17 }
18
19 @Override
20 public void Error(String Error){
21     StringBuilder result = new StringBuilder();
22     for (int i = 0; i < Error.length(); i++) {
23         if (i > 0) {
24             result.append(" ");
25         }
26         result.append(Error.charAt(i));
27     }
28     System.out.println("ERROR: " + result.toString());
29 }
```

Problems Javadoc Console <terminated> App (1) [Java Application] C:\Users\w\ind\p2\pool\plugins\org.eclipse.justj.openj

```
***Hello***
*****
***ERROR: Hello***
*****
H e l l o
ERROR: H e l l o
```

Steps 9, 10, & 11 (App):



The screenshot shows the Eclipse IDE with the 'App.java' file open. The code defines a public class 'App' with a static 'main' method. Inside 'main', two logger objects are instantiated: 'AsteriskLogger' and 'SpacedLogger'. The 'AsteriskLogger' is used to log 'Hello' and an error 'Hello'. The 'SpacedLogger' is used to log 'Hello' and an error 'Hello'. The console output at the bottom shows the results of these log statements: '***Hello***' for the first log, '***ERROR: Hello***' for the first error, 'H e l l o' for the second log, and 'ERROR: H e l l o' for the second error.

```
1 public class App {
2
3     public static void main(String[] args) {
4
5         Logger logger = new AsteriskLogger();
6         Logger logger2 = new SpacedLogger();
7
8         logger.log("Hello");
9         logger.Error("Hello");
10
11        logger2.log("Hello");
12        logger2.Error("Hello");
13    }
14 }
15
16
17
18
19
```

Problems @ Javadoc Console

<terminated> App (1) [Java Application] C:\Users\w\ind\p2\pool\plugins\org.eclipse.
Hello

ERROR: Hello

H e l l o
ERROR: H e l l o

URL to GitHub Repository:

<https://github.com/wlindstrom55/week-5-assignments>