Contents

New API Logging SE9

Introduce

This API has been introduced in java to provide a common mechanism to handle all the platform logs and to expose a service interface that can be customized by libraries and applications. This way, the JDK platform logs can use the same logging framework as the application, and the project dependencies can be reduced.

Use Logger Default Work

Simple demo with print log into Consolog
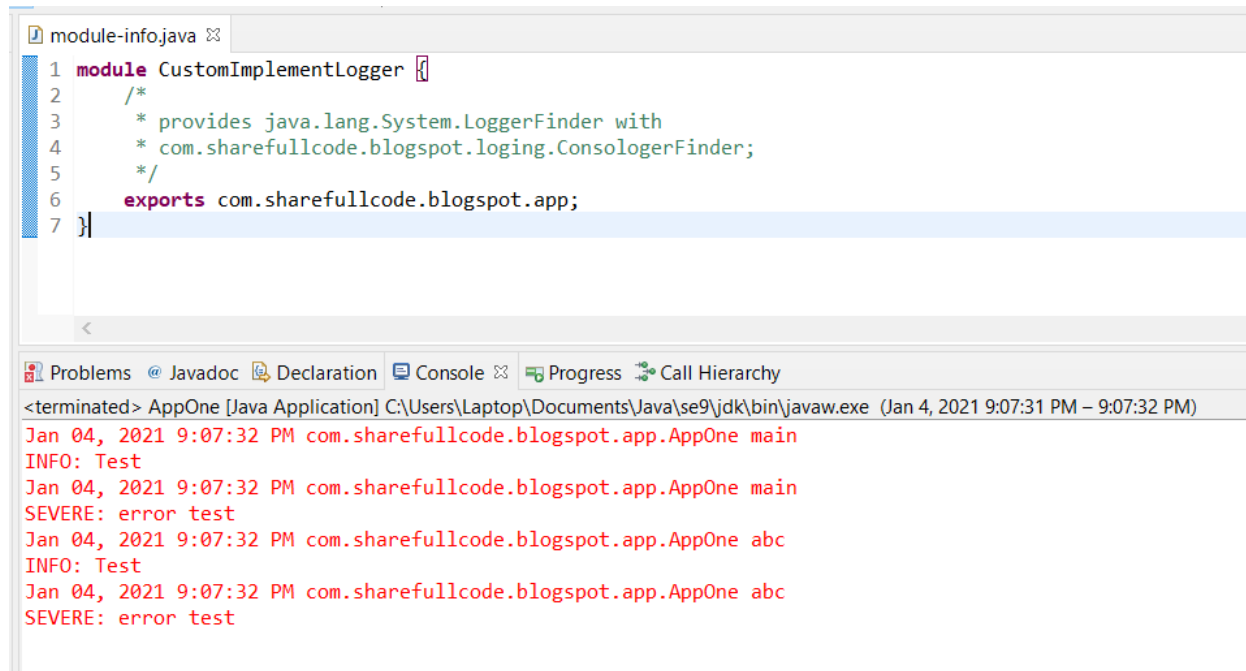
Create Logger inplement the System.Logger

```java
 1  package com.sharefullcode.blogspot.loging;
 2
 3  import java.text.MessageFormat;
 4  import java.util.ResourceBundle;
 5
 6  public class Consologer implements System.Logger{
 7
 8      @Override
 9      public String getName() {
10          return "Consolog name";
11      }
12
13      @Override
14      public boolean isLoggable(Level level) {
15          return true;
16      }
17
18      @Override
19      public void log(Level level, ResourceBundle bundle, String msg, Throwable thrown) {
20          System.out.printf("ConsoleLogger [%s]: %s - %s%n", level, msg, thrown);
21      }
22
23      @Override
24      public void log(Level level, ResourceBundle bundle, String format, Object... params) {
25          System.out.printf("ConsoleLogger [%s]: %s%n", level,
26                  MessageFormat.format(format, params));
27      }
28
29  }
30
```

We need to implement a LoggerFinder that creates instances of our Consologer.

```java
 1  package com.sharefullcode.blogspot.loging;
 2
 3  import java.lang.System.Logger;
 4
 5  public class ConsologerFinder extends System.LoggerFinder{
 6
 7      @Override
 8      public Logger getLogger(String name, Module module) {
 9          return new Consologer();
10      }
11
12  }
13
```

Package Explorer:
- commonfeaturesse9
- CustomImplementLogger
  - src
    - com.sharefullcode.blogspot
      - app
        - AppOne.java
      - loging
        - Consologer.java
        - ConsologerFinder.java
  - module-info.java
- JRE System Library [JavaSE-9]

Finally, we need to register out LoggerFinder as a Service so it can be discovered by JDK. If we don't provide an implementation, the SimpleConsoleLogger will be used by default.
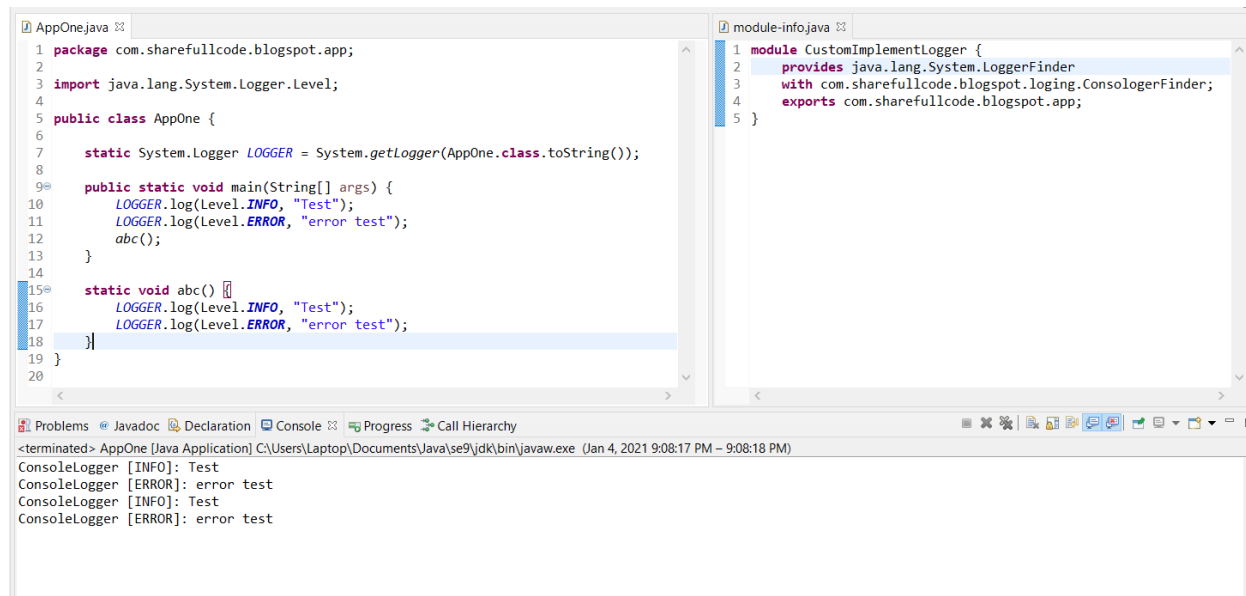
```
module-info.java ⊠
1  module CustomImplementLogger {
2      /*
3       * provides java.lang.System.LoggerFinder with
4       * com.sharefullcode.blogspot.loging.ConsologerFinder;
5       */
6      exports com.sharefullcode.blogspot.app;
7  }
```

Problems @ Javadoc 🔍 Declaration 🖥 Console ⊠ 🔧 Progress 📞 Call Hierarchy

<terminated> AppOne [Java Application] C:\Users\Laptop\Documents\Java\se9\jdk\bin\javaw.exe (Jan 4, 2021 9:07:31 PM – 9:07:32 PM)

```
Jan 04, 2021 9:07:32 PM com.sharefullcode.blogspot.app.AppOne main
INFO: Test
Jan 04, 2021 9:07:32 PM com.sharefullcode.blogspot.app.AppOne main
SEVERE: error test
Jan 04, 2021 9:07:32 PM com.sharefullcode.blogspot.app.AppOne abc
INFO: Test
Jan 04, 2021 9:07:32 PM com.sharefullcode.blogspot.app.AppOne abc
SEVERE: error test
```

```
AppOnejava ⊠
1  package com.sharefullcode.blogspot.app;
2
3  import java.lang.System.Logger.Level;
4
5  public class AppOne {
6
7      static System.Logger LOGGER = System.getLogger(AppOne.class.toString());
8
9      public static void main(String[] args) {
10         LOGGER.log(Level.INFO, "Test");
11         LOGGER.log(Level.ERROR, "error test");
12         abc();
13     }
14
15     static void abc() {
16         LOGGER.log(Level.INFO, "Test");
17         LOGGER.log(Level.ERROR, "error test");
18     }
19 }
20
```

```
module-info.java ⊠
1  module CustomImplementLogger {
2      provides java.lang.System.LoggerFinder
3      with com.sharefullcode.blogspot.loging.ConsologerFinder;
4      exports com.sharefullcode.blogspot.app;
5  }
```
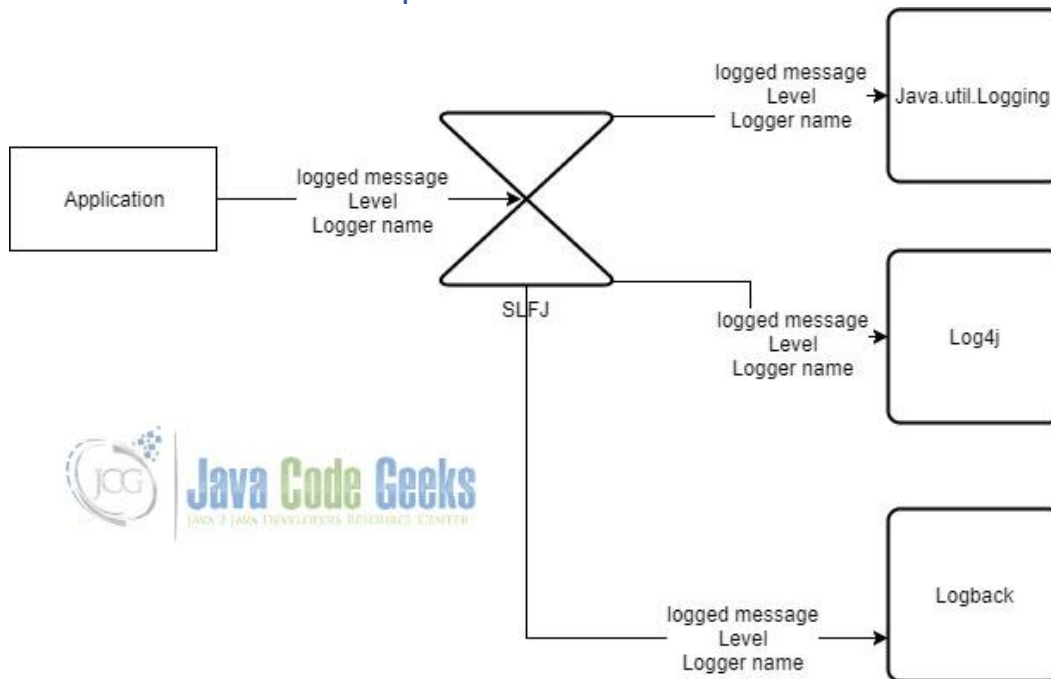
Problems @ Javadoc 🔍 Declaration 🖥 Console ⊠ 🔧 Progress 📞 Call Hierarchy

<terminated> AppOne [Java Application] C:\Users\Laptop\Documents\Java\se9\jdk\bin\javaw.exe (Jan 4, 2021 9:08:17 PM – 9:08:18 PM)

```
ConsoleLogger [INFO]: Test
ConsoleLogger [ERROR]: error test
ConsoleLogger [INFO]: Test
ConsoleLogger [ERROR]: error test
```

Use Extenal Logging Framwork [Log4j2]

One of the most useful uses of the Logging API in java 9 is to let applications route the JDK logs to the same logging framework the application is using.

Custom Framworks implement use SLF4J

https://examples.javacodegeeks.com/enterprise-java/slf4j/slf4j-tutorial-beginners/

Add File pom.xml

```xml
M *DemoApiLoggingWithLog4j2/pom.xml ☒
 1⊖ <project xmlns="http://maven.apache.org/POM/4.0.0"
 2     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
 4     <modelVersion>4.0.0</modelVersion>
 5     <groupId>sharefullcode</groupId>
 6     <artifactId>DemoApiLoggingWithLog4j2</artifactId>
 7     <version>0.0.1-SNAPSHOT</version>
 8
 9⊖    <dependencies>
10
11         <!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-api -->
12⊖        <dependency>
13             <groupId>org.slf4j</groupId>
14             <artifactId>slf4j-api</artifactId>
15             <version>1.7.30</version>
16         </dependency>
17
18         <!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-log4j12 -->
19⊖        <dependency>
20             <groupId>org.slf4j</groupId>
21             <artifactId>slf4j-log4j12</artifactId>
22⊖            <version>1.7.30</version>
23         </dependency>
24
25         <!-- https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-core -->
26⊖        <dependency>
27             <groupId>org.apache.logging.log4j</groupId>
28             <artifactId>log4j-core</artifactId>
29             <version>2.14.0</version>
30         </dependency>
31     </dependencies>
32
33 </project>
```

Create Logging implement System

```java
1  package com.sharefullcode.blogspot;
2
3  import java.util.ResourceBundle;
4
5  import org.slf4j.Logger;
6  import org.slf4j.LoggerFactory;
7
8  public class Slf4jCustomImplementation implements System.Logger{
9
10     private String name;
11     private Logger loggerSlf4j;
12
13     public Slf4jCustomImplementation(String name) {
14         this.name = name;
15         this.loggerSlf4j = LoggerFactory.getLogger(name);
16     }
17
```

The method remain is implement

```java
16     }
17
18     public String getName() {
19         return name;
20     }
21
22     public boolean isLoggable(Level level) {
23         switch (level) {
24         case OFF:
25             return false;
26         case TRACE:
27             return loggerSlf4j.isTraceEnabled();
28         case DEBUG:
29             return loggerSlf4j.isDebugEnabled();
30         case INFO:
31             return loggerSlf4j.isInfoEnabled();
32         case WARNING:
33             return loggerSlf4j.isWarnEnabled();
34         case ERROR:
35             return loggerSlf4j.isErrorEnabled();
36         case ALL:
37         default:
38             return true;
39         }
40     }
41
42     public void log(Level level, ResourceBundle bundle, String msg, Throwable thrown) {
43         // TODO Auto-generated method stub
44
45     }
46
47     public void log(Level level, ResourceBundle bundle, String format, Object... params) {
48         // TODO Auto-generated method stub
49
50     }
```

```
67          }
68      }
69
70⊖    public void log(Level level, ResourceBundle bundle, String format, Object... params) {
71          if (!isLoggable(level)) {
72              return;
73          }
74          String msg = MessageFormat.format(format, params);
75
76          switch (level) {
77          case TRACE:
78              loggerSlf4j.trace(msg);
79              break;
80          case DEBUG:
81              loggerSlf4j.debug(msg);
82              break;
83          case INFO:
84              loggerSlf4j.info(msg);
85              break;
86          case WARNING:
87              loggerSlf4j.warn(msg);
88              break;
89          case ERROR:
90              loggerSlf4j.error(msg);
91              break;
92          case ALL:
93          default:
94              loggerSlf4j.info(msg);
95          }
96      }
97 }
```

Create System.FinderLogger

```java
1 package com.sharefullcode.blogspot;
2
3 import java.lang.System.Logger;
4
5 public class Slf4jLoggerFinder extends System.LoggerFinder{
6
7⊖    @Override
8      public Logger getLogger(String name, Module module) {
9          return new Slf4jCustomImplementation(name);
10     }
11 }
12
```
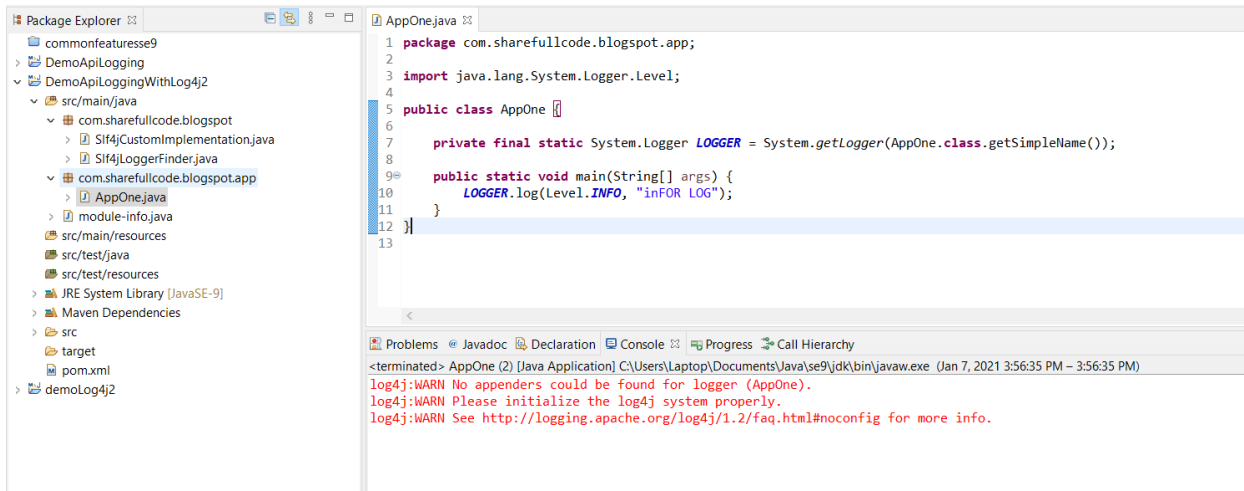
Add into moudule-infor.java
provides ... with ... Register service in the module-info
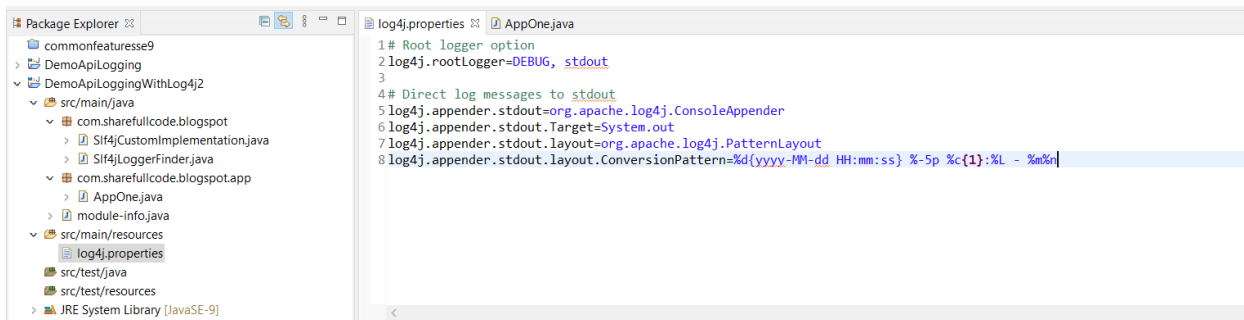
```java
1 module DemoApiLoggingWithLog4j2 {
2     exports com.sharefullcode.blogspot.app;
3     exports com.sharefullcode.blogspot;
4
5     provides java.lang.System.LoggerFinder with
6     com.sharefullcode.blogspot.Slf4jLoggerFinder;
7
8     requires org.slf4j;
9 }
```
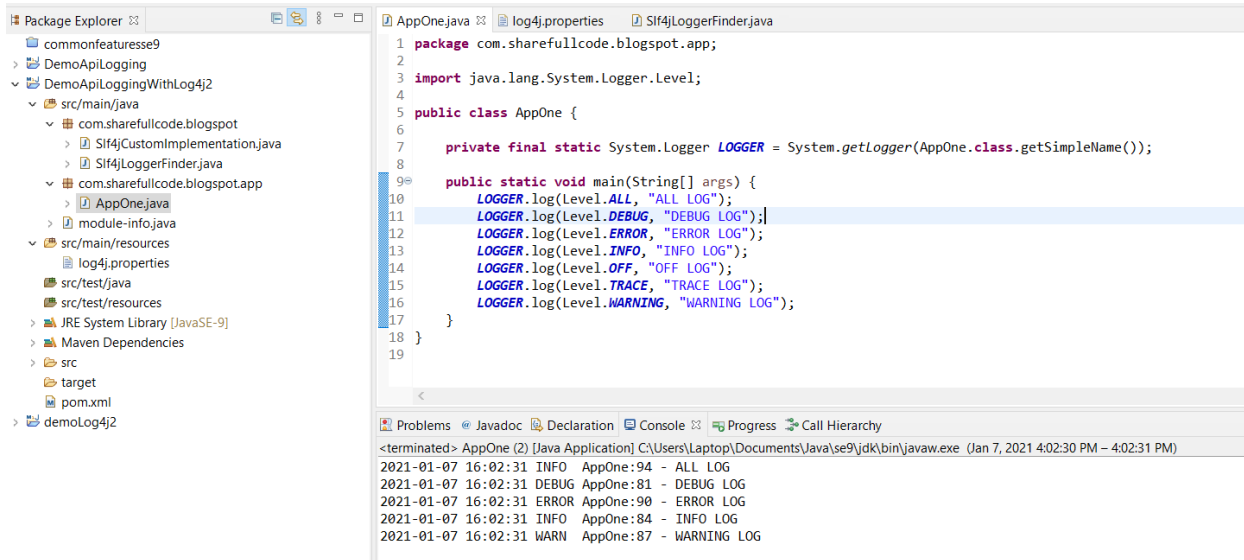
Not yet create file log4j config

Src/resources/Log4j.properties

Ok Run Done

Reference

This section is non commercial mainly sharing and advance knowlage for java.This tutorials has referenced document from the list below if you has complain for license, i will remove all from internet. Thank you all everything.

https://www.baeldung.com/java-9-logging-api

https://examples.javacodegeeks.com/enterprise-java/slf4j/slf4j-tutorial-beginners/