**Spring Boot Logging with Lombok**

Project Lombok is very handy tool for removing the boilerplate code from application. Lombok can also be used to configure logging in spring boot applications and thus removing the boilerplate code for getting the logger instance.

**1. Import lombok in spring boot application**

Before using it, we must import the lombok in spring boot application. Don’t forget to install lombok into eclipse before using it.

|  |
| --- |
| pom.xml |
| <project xmlns="<http://maven.apache.org/POM/4.0.0>"      xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"      xsi:schemaLocation="<http://maven.apache.org/POM/4.0.0>  <http://maven.apache.org/xsd/maven-4.0.0.xsd>">      <modelVersion>4.0.0</modelVersion>        <groupId>com.company</groupId>      <artifactId>SpringBootEhcache</artifactId>      <version>0.0.1-SNAPSHOT</version>      <packaging>jar</packaging>        <name>SpringBootEhcache</name>      <url>[http://maven.apache.org](http://maven.apache.org/)</url>        <parent>          <groupId>org.springframework.boot</groupId>          <artifactId>spring-boot-starter-parent</artifactId>          <version>2.1.6.RELEASE</version>          <relativePath />      </parent>        <properties>          <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>          <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>          <java.version>1.8</java.version>          <skipTests>true</skipTests>      </properties>        <dependencies>            <dependency>              <groupId>org.springframework.boot</groupId>              <artifactId>spring-boot-starter-web</artifactId>          </dependency>            <dependency>              <groupId>org.projectlombok</groupId>              <artifactId>lombok</artifactId>          </dependency>        </dependencies>        <build>          <plugins>              <plugin>                  <groupId>org.springframework.boot</groupId>                  <artifactId>spring-boot-maven-plugin</artifactId>              </plugin>          </plugins>      </build>  </project> |

**2. @Log, @Log4j2, @Slf4j – Lombok annotations**

Instead, we can start writing log statements in a java class which is annotated with lombok’s @Log annotations. Lombok supports following log annotations for spring boot –

* @CommonsLog – Creates the logger with following statement:

|  |
| --- |
| private static final org.apache.commons.logging.Log log =      org.apache.commons.logging.LogFactory.getLog(LogExample.class); |

* @Log – Creates the logger with following statement:

|  |
| --- |
| private static final java.util.logging.Logger log =      java.util.logging.Logger.getLogger(LogExample.class.getName()); |

* @Log4j2 – Creates the logger with following statement:

|  |
| --- |
| private static final org.apache.logging.log4j.Logger log =      org.apache.logging.log4j.LogManager.getLogger(LogExample.class); |

* @Slf4j – Creates the logger with following statement:

|  |
| --- |
| Creates private static final org.slf4j.Logger log =      org.slf4j.LoggerFactory.getLogger(LogExample.class); |

A sample usage of this log annotation is :

|  |
| --- |
| Application.java |
| import org.springframework.boot.SpringApplication;  import org.springframework.boot.autoconfigure.SpringBootApplication;  import lombok.extern.slf4j.Slf4j;    @Slf4j  @SpringBootApplication  public class Application  {      public static void main(String[] args) {          SpringApplication.run(Application.class, args);           //Note here we have not declare          //any Logger object explicitly, it get generated by Lombok //automatically with above annotation          log.info("Simple log statement with inputs {}, {} and {}", 1, 2, 3);      }  } |

By default, spring boot uses logback as logging provider.

**3. Using Log4j2 Logging with Lombok**

**To use log4j2, exclude logback from classpath and include log4j2.**

|  |
| --- |
| pom.xml |
| <dependencies>        <dependency>          <groupId>org.springframework.boot</groupId>          <artifactId>spring-boot-starter-web</artifactId>          <exclusions>              <exclusion>                  <groupId>org.springframework.boot</groupId>                  <artifactId>spring-boot-starter-logging</artifactId>              </exclusion>          </exclusions>      </dependency>        <dependency>          <groupId>org.springframework.boot</groupId>          <artifactId>spring-boot-starter-log4j2</artifactId>      </dependency>        <dependency>          <groupId>org.projectlombok</groupId>          <artifactId>lombok</artifactId>      </dependency>    </dependencies> |

Now you can use either @Slf4j (**recommneded**) or @Log4j2 at class to use log4j2 as underlying logging implementation.

|  |
| --- |
| Application.java |
| import lombok.extern.slf4j.Slf4j;    @Slf4j  @SpringBootApplication  public class Application  {      public static void main(String[] args) {//Note here we have not declare          //any Logger object explicitly, it get generated by Lombok //automatically with above annotation          SpringApplication.run(Application.class, args);          log.info("Simple log statement with inputs {}, {} and {}", 1, 2, 3);      }  } |
| Console | |
| 2019-07-28 15:29:41.267  INFO 7660 --- [main] com.javaLive.demo.Application      : Simple log statement with inputs 1, 2 and 3 | |