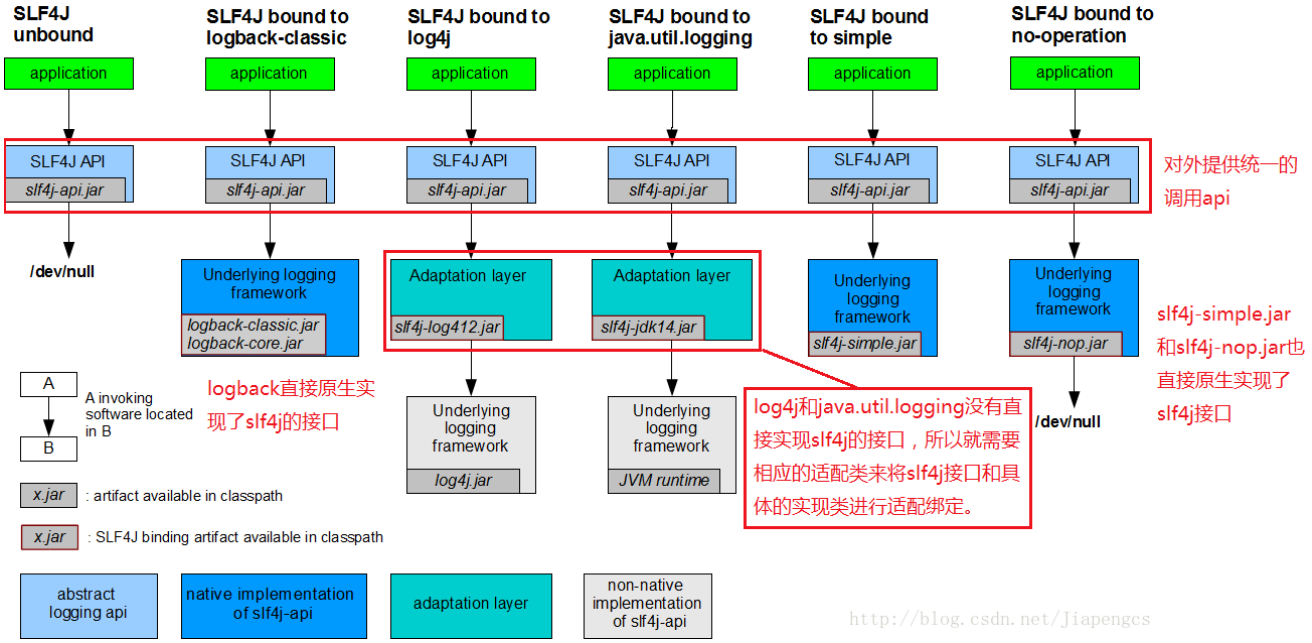
1.SLF4J(Simple logging Facade for Java)

意思为简单日志门面，它是把不同的日志系统的实现进行了具体的抽象化，只提供了统一的日志使用接口，使用时只需要按照其提供的接口方法进行调用即可，由于它只是一个接口，并不是一个具体的可以直接单独使用的日志框架，所以最终日志的格式、记录级别、输出方式等都要通过接口绑定的具体的日志系统来实现，这些具体的日志系统就有log4j,logback,java.util.logging等，它们才实现了具体的日志系统的功能。

如何使用SLF4J?

既然SLF4J只是一个接口，那么实际使用时必须要结合具体的日志系统来使用，我们首先来看SLF4J和各个具体的日志系统进行绑定时的框架原理图：



其实slf4j原理很简单，他只提供一个核心slf4j api(就是slf4j-api.jar包)，这个包只有日志的接口，并没有实现，所以如果要使用就得再给它提供一个实现了些接口的日志包，比 如：log4j,common logging,jdk log日志实现包等，但是这些日志实现又不能通过接口直接调用，实现上他们根本就和slf4j-api不一致，因此slf4j又增加了一层来转换各日志实现包的使 用，当然slf4j-simple除外。其结构如下：

slf4j-api(接口层)

|

各日志实现包的连接层(slf4j-jdk14, slf4j-log4j)

|

各日志实现包

所以，结合各日志实现包使用时提供的jar包情况为：

SLF4J和logback结合使用时需要提供的jar:slf4j-api.jar,logback-classic.jar,logback-core.jar

SLF4J和log4j结合使用时需要提供的jar:slf4j-api.jar,slf4j-log412.jar,log4j.jar

SLF4J和JDK中java.util.logging结合使用时需要提供的jar:slf4j-api.jar,slf4j-jdk14.jar
SLF4J和simple(SLF4J本身提供的一个接口的简单实现)结合使用时需要提供的jar:slf4j-api.jar,slf4j-simple.jar
当然还有其他的日志实现包，以上是经常会使用到的一些。
注意，以上slf4j和各日志实现包结合使用时最好只使用一种结合，不然的话会提示重复绑定日志，并且会导致日志无法输出。

为什么要使用SLF4J?

- slf4j是一个日志接口，自己没有具体实现日志系统，只提供了一组标准的调用api,这样将调用和具体的日志实现分离，使用slf4j后有利于根据自己实际的需求更换具体的日志系统，比如，之前使用的具体的日志系统为log4j,想更换为logback时，只需要删除log4j相关的jar,然后加入logback相关的jar和日志配置文件即可，而不需要改动具体的日志输出方法，试想如果没有采用这种方式，当你的系统中日志输出有成千上万条时，你要更换日志系统将是多么庞大的一项工程。如果你开发的是一个面向公众使用的组件或公共服务模块，那么一定要使用slf4的这种形式，这有利于别人在调用你的模块时保持和他系统中使用统一的日志输出。
- slf4j日志输出时可以使用{}占位符，如，logger.info("testlog: {}", "test")，而如果只使用log4j做日志输出时，只能以logger.info("testlog:"+"test")这种形式，前者要比后者在性能上更好，后者采用+连接字符串时就是new 一个String 字符串，在性能上就不如前者。

2.log4j(log for java)

Log4j是Apache的一个开源项目，通过使用Log4j，我们可以控制日志信息输送的目的地是控制台、文件、GUI组件，甚至是套接口服务器、NT的事件记录器、UNIX Syslog守护进程等；我们也可以控制每一条日志的输出格式；通过定义每一条日志信息的级别，我们能够更加细致地控制日志的生成过程。最令人感兴趣的就是，这些可以通过一个配置文件来灵活地进行配置，而不需要修改应用的代码。

如何使用?

- 引入jar,使用log4j时需要的jar为：log4j.jar。
- 定义配置文件log4j.properties或log4j.xml
- 在具体的类中进行使用:
    - 在需要日志输出的类中加入：private static final Logger logger = Logger.getLogger(Tester.class); //通过Logger获取Logger实例
    - 在需要输出日志的地方调用相应方法即可：logger.debug(“System …..”); logger.log(Level.DEBUG,"这是debug");// 可以指定打印级别,但logback不支持,需要自己实现

关于如何单独使用log4j，建议详细阅读以下文章:

https://blog.csdn.net/u012422446/article/details/51199724
https://blog.csdn.net/azheng270/article/details/2173430/
http://shmilyaw-hotmail-com.iteye.com/blog/2410764

3.logback

logback同样是由log4j的作者设计完成的，拥有更好的特性，用来取代log4j的一个日志框架,是slf4j的原生实现(即直接实现了slf4j的接口，而log4j并没有直接实现，所以就需要一个适配器slf4j-log4j12.jar),logback一共有以下几个模块：

- logback-core：其它两个模块的基础模块
- logback-classic：它是log4j的一个改良版本，同时它完整实现了slf4j API使你可以很方便地更换成其它日志系统如log4j或JDK14 Logging
- logback-access：访问模块与Servlet容器集成提供通过Http来访问日志的功能

同样，单独使用它时，需要引入以上jar,然后进行配置文件的配置，最后就是在相关类中进行使用，使用时加入以下语句：

```
private final static Logger logger = LoggerFactory.getLogger(Test.class);
 logger.info("打印日志");
```

对于logback的使用，详细使用方法及配置推荐阅读以下文章：
https://www.cnblogs.com/warking/p/5710303.html

springboot 默认使用框架是logback

4.总结如下：

1、slf4j是java的一个日志门面，实现了日志框架一些通用的api，log4j和logback是具体的日志框架。
2、他们可以单独的使用，也可以绑定slf4j一起使用。

单独使用，分别调用框架自己的方法来输出日志信息。绑定slf4j一起使用。调用slf4j的api来输入日志信息，具体使用与底层日志框架无关（需要底层框架的配置文件）。显然不推荐单独使用日志框架。假设项目中已经使用了log4j，而我们此时加载了一个类库，而这个类库依赖另一个日志框架。这个时候我们就需要维护两个日志框架，这是一个非常麻烦的事情。而使用了slf4j就不同了，由于应用调用的抽象层的api，与底层日志框架是无关的，因此可以任意更换日志框架。