**MyBatis Generator代码分析二（日志）**

[](http://www.jianshu.com/u/231b43e2c05f)

[小码哥Java学院](http://www.jianshu.com/u/231b43e2c05f) 关注

2015.09.06 11:00 字数 494 阅读 709评论 1喜欢 2

**org.mybatis.generator.internal.util.messages.Message类：**

用于获得消息做国际化，使用的bundle是org.mybatis.generator.internal.util.messages.messages；  
小技巧，在输出大量信息的时候（比如help消息）

* 先创建一个key:Usage.Lines=27,得到一共多少行消息，
* 然后用for循环遍历每一行消息输出：Usage.0~Usage.27

**org.mybatis.generator.logging:**

放的是MGB日志相关内容；在MBG中，日志支持两种，一种是Log4J，一种是JDK的logging，而这个包里面的类就是对日志的一个门面。我的理解，在MBG出来的时候，还没有slf4j，所以没有用slf4j，感觉这个包其实比较没太大意义；

* AbstractLogFactory:一个日志工厂的接口，提供了一个getLog(Class)方法对传入的Class返回一个Log；
* Log:对日志器的接口（具体Logger的门面）；提供了通用的日志器方法：
  + isDebugEnabled：是否debug级别；
  + error：记录错误
  + debug：记录debug信息；
  + warn：记录错误信息；
* Log4jImpl:对Log4J的一个适配；简单对比一个方法：
* public JdkLoggingImpl(Class<?> clazz) {
* log = Logger.getLogger(clazz.getName());
* }
* public boolean isDebugEnabled() {
* return log.isLoggable(Level.FINE);
* }
* public void error(String s, Throwable e) {
* LogRecord lr = new LogRecord(Level.SEVERE, s);
* lr.setSourceClassName(log.getName());
* lr.setThrown(e);
* log.log(lr);

}

* JdkLoggingImpl:对JDK的logging的适配；简单的看一个方法就明白：
* public Log4jImpl(Class<?> clazz) {
* log = Logger.getLogger(clazz);
* }
* public boolean isDebugEnabled() {
* return log.isDebugEnabled();
* }
* public void error(String s, Throwable e) {
* log.error(s, e);

}

* LogFactory:对Log工厂的一个包装；在LogFactory中持有一个AbstractLogFactory引用；  
  在LogFactory中持有两个实现了AbstractLogFactory的静态内部类：
  + JdkLoggingLogFactory：对JDKlogging的包装类，其创建的Logger其实就是JdkLoggingImpl；
  + Log4jLoggingLogFactory：对Log4J的包装类，其创建的Logger其实就是Log4jImpl；

在MBG中，都是使用LogFactory的LogFactory.getLog(Class targetLogClass)直接获取对一个Class的日志对象；  
这个日志工厂在MBG中是固定的，不容易直接扩展；