1. logback日志系统(<https://logback.qos.ch/manual> )

### Conditional processing of configuration files

Developers often need to juggle between several logback configuration files targeting different environments such as development, testing and production. These configuration files have substantial parts in common differing only in a few places. To avoid duplication, logback supports conditional processing of configuration files with the help of <if>, <then> and <else> elements so that a single configuration file can adequately target several environments. Note that conditional processing requires the [Janino library](https://logback.qos.ch/setup.html#janino).

The general format for conditional statements is shown below.

   <!-- if-then form -->  
   <if condition="some conditional expression">  
    <then>  
      ...  
    </then>  
  </if>  
    
  <!-- if-then-else form -->  
  <if condition="some conditional expression">  
    <then>  
      ...  
    </then>  
    <else>  
      ...  
    </else>      
  </if>

The condition is a Java expression in which only context properties or system properties are accessible. For a key passed as argument, the property() or its shorter equivalent p() methods return the String value of the property. For example, to access the value of a property with key "k", you would write property("k") or equivalently p("k"). If the property with key "k" is undefined, the property method will return the empty string and not null. This avoids the need to check for null values.

The isDefined() method can be used to check whether a property is defined. For example, to check whether the property "k" is defined you would write isDefined("k") Similarly, if you need to check whether a property is null, the isNull() method is provided. Example: isNull("k").

<configuration debug="true">  
  
  **<if condition='property("HOSTNAME").contains("torino")'>**  
    **<then>**  
      <appender name="CON" class="ch.qos.logback.core.ConsoleAppender">  
        <encoder>  
          <pattern>%d %-5level %logger{35} - %msg %n</pattern>  
        </encoder>  
      </appender>  
      <root>  
        <appender-ref ref="CON" />  
      </root>  
    **</then>**  
  **</if>**  
  
  <appender name="FILE" class="ch.qos.logback.core.FileAppender">  
    <file>${randomOutputDir}/conditional.log</file>  
    <encoder>  
      <pattern>%d %-5level %logger{35} - %msg %n</pattern>  
   </encoder>  
  </appender>  
  
  <root level="ERROR">  
     <appender-ref ref="FILE" />  
  </root>  
</configuration>

Conditional processing is supported anywhere within the <configuration> element. Nested if-then-else statements are also supported. However, XML syntax is awfully cumbersome and is ill suited as the foundation of a general purpose programming language. Consequently, too many conditionals will quickly render your configuration files incomprehensible to subsequent readers, including yourself.

1. git生产／测试