

第五节 配置管理与 Java Logging

配置管理

设计技巧

优先读取外部配置（存在多来源，可能存在优先级），将内部配置作为默认（兜底）策略

一手微信study322 爱学习都加我

当存在多源外部配置时，源作用范围越小，通常越优先，比如 Java System#getProperties()（当前 JVM 进程）和操作系统环境变量（当前用户或者系统共享）

Java SE 标准外部化配置

java.util.prefs.Preferences

树形层次接口，类似于目录

用户偏好数据

Windows - 注册表实现

Linux/Unix - 使用文件实现

主要方法

`get(String,String)`

`getDouble(String,double)`

`getFloat(String,float)`

...

具备类型转换

Java EE 标准外部化配置

一手微信study522 爱学习都加我

Servlet 上下文配置 -

`javax.servlet.ServletContext#getInitParameter(String)`

Servlet 配置 -

`javax.servlet.ServletConfig#getInitParameter(String)`

JSP 配置 - `javax.servlet.descriptor.JspConfigDescriptor`

JNDI 配置 -
javax.naming.Context#lookup(javax.naming.Name)

关联技术

Java SE 属性配置和类型转换 API

一手微信study322 爱学习都加我

类型	属性来源	API
Integer、int	Java System Properties	java.lang.Integer#getInteger
Long、long	Java System Properties	java.lang.Long#getLong

类型	属性来源	API
Boolean、boolean	Java System Properties	java.lang.Boolean#getBoolean

整合 MicroProfile Config

Maven 依赖 一手微信study322 爱学习都加我

```
<dependency>
  <groupId>org.eclipse.microprofile.config</groupId>
  <artifactId>microprofile-config-api</artifactId>
  <version>2.0</version>
</dependency>
```

配置统一门面接口 -

org.eclipse.microprofile.config.Config

- 与 配置来源 - ConfigSource 一对多
- 相较于 ConfigSource 获取配置而言，它增加了类型转换

配置来源 - ConfigSource

- 属性键值对信息 - `getProperties()`
- 当前配置的绝对顺序 - `getOrdinal()`
- 获取配置方法 - `getValue()`

配置 SPI -

`org.eclipse.microprofile.config.spi.ConfigProviderResolver`

利用 Java ServiceLoader 来加载对应的实现类

一手微信 study322 爱学习都加我

需求

1. 应用程序仅知道配置名称，不关心数据来源
 - 属性名称： `application.name`
 - ordinal 1 - `ServletConfig`
 - ordinal 2 - `ServletContext`
 - ordinal 3 - `JNDI`
 - ordinal 4 - `System#getProperties()`
 - ordinal 5 - OS 环境变量
 - ordinal 6 - 文件
 - `Config#getValue("application.name") = ?`
2. 实现配置监听变化

Java Logging

从 Java 1.4 引入的，部分设计参考了 Log4j 设计

日志数据类型 - LogRecord

在 log4j 里面命名为 LogEvent

日志过滤器 - Filter

决定当前的 LogRecord 是否要输出

日志处理对象 - Handler

官方文档

<https://docs.oracle.com/javase/8/docs/technotes/guides/logging/index.html>