链滴

# JAVA 实践项目 --- 树莓派信息自动化采集后入库项目 (六)

作者：HuixiaZhang

原文链接：https://ld246.com/article/1539941240065

来源网站：链滴

项目源代码可访问我的github：https://github.com/Spacider/Gather-and-store
如果觉得好的话请给个star哦~

开发IDE： IDEA 2018.03 JDK 1.8
开发环境： macOS 10.13.6 （如windows请对项目中部分路径进行改写）
数据库: Oracle 11g

---

上节完毕项目主体已经基本开发完成，下列进行一系列优化操作：

1.备份模块：
先编写一个接口：

```
/**
 * 备份模块
 * 1. 客户端发送集合没有发送出去或连不上服务器需要备份
 * 2. 服务端接收了集合对象写入数据库出错，对集合进行备份
 */
public interface BackUp extends WossModel{

    /**
     * 将集合保存到文件中
     * @param coll 需要备份的集合
     */
    void storeEnvs(Collection<Environment> coll );


    /**
     * 将集合从文件中读取
     * @return
     */
    Collection<Environment> loadEnvs();
}
```

具体实现：
通过对象流套文件流的形式把对象存入备份文件中：

```
@Override
public void storeEnvs(Collection<Environment> coll) {
    File file = new File(path);
    FileOutputStream fos = null;
    ObjectOutputStream oos = null;
    try {
        if (!file.exists()) {
            file.createNewFile();
        }
        fos = new FileOutputStream(file,true);
        oos = new ObjectOutputStream(fos);
        oos.writeObject(coll);
        oos.flush();
        System.out.println("已经存入备份文件中 ,path:" +path);
```

```
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        IOUtil.close(fos,oos);
    }
}
```

从备份文件中取出相应对象！

```
@Override
public Collection <Environment> loadEnvs() {
    FileInputStream fis =  null;
    ObjectInputStream ois = null;
    Object backupObject = null;

    try {
        fis = new FileInputStream(path);
        ois = new ObjectInputStream(fis);

        backupObject = ois.readObject();

//            System.out.println("已经从 path:" + path + "取出备份文件");

        log.info("从 path:" + path + "取出备份文件中...");
    } catch (IOException | ClassNotFoundException e) {
        e.printStackTrace();
    }
    return (Collection <Environment>) backupObject;
}
```

---

2.日志模块

先编写一个接口叫Log，里边封装了日志的四种级别：

```
public interface Log extends WossModel{

    /**
     * 输出 debug 级别的日志
     * @param msg
     */
    void debug(String msg);

    /**
     * 输出 info 级别的日志
     * @param msg
     */
    void info(String msg);

    /**
     * 输出 warn 级别的日志
     * @param msg
     */
    void warn(String msg);
```

```
    /**
     * 输出 error 级别的日志
     * @param msg
     */
    void error(String msg);

}
```

日志模块的实现就是调用log4j的日志配置，然后做对应的输出：

```
    private Logger LOGGER = Logger.getLogger(LogImpl.class);

    public LogImpl() {
        PropertyConfigurator.configure("/Users/wjh/Desktop/FirstProject/src/main/resources/lo
4j.properties");
    }

    @Override
    public void debug(String msg) {
        LOGGER.debug(msg);
    }

    @Override
    public void info(String msg) {
        LOGGER.info(msg);
    }

    @Override
    public void warn(String msg) {
        LOGGER.warn(msg);
    }

    @Override
    public void error(String msg) {
        LOGGER.error(msg);
    }

    @Override
    public void init(Properties properties) {

    }
```

---

3.配置模块

首先配置一个接口为WossModel，里面有一个方法,有了这个方法之后当一个类如果实现了这个接口我们就可以把Properties文件传给它，在Properties文件中我们又可以传入相应的参数，这样做可以活替换类中的可变参，让整个程序变得灵活起来！

```
/**
 * 初始化配置参数
```

```
 */
public interface WossModel {
    void init(Properties properties);
}
```

再定义一个接口为Configuration,在这个接口的实现中会通过反射去获取各个类(又可以说是一个模块的对象，保证了对象的唯一性！这一部分又可以看做是一个注入过程，让对象的创建变得灵活，变得全!

```
/**
 * 获取各个配置模块对象
 * 对所有对象进行管理
 */
public interface Configuration {
    /**
     * 获取采集模块对象
     */
    Gather getGather();

    /**
     * 获取客户端模块对象
     */
    EnvClient getClient();

    /**
     * 获取服务器端对象
     */
    EnvServer getServer();

    /**
     * 获取入库模块对象
     */
    DBStore getDBStore();

    /**
     * 获取日志模块对象
     */
    Log getLog();

    /**
     * 获取备份模块对象
     */
    BackUp getBackUp();

}
```

最后一个接口是ConfigurationAware，这个接口的主要用途就是当遇到一个模块对象调用到另一个块对象时间，就可以把刚才写的Configuration整个注入进去，有了这个Configuration对象就相当于以获取到了其他所有配置模块的对象，这样你就可以灵活去获取到你需要的对象并且调用其内的方法!

```
public interface ConfigurationAware {
```

```
    void SetConfiguration(Configuration conf);

}
```

## 具体实施

讲了这么多，具体咋实施呢：

给你需要的类上接口！

例如：

客户端的类：

让接口实现WossModel接口，那么子实现就默认也实现了这个接口！

由于在客户端类中

```
public interface EnvClient extends WossModel {
}
public class EnvClientImpl implements EnvClient , ConfigurationAware {
}
```

当继承了WossModel接口之后我们就可以实现其方法来接入Properties文件:

```
public void init(Properties properties) {
    port = Integer.parseInt(properties.getProperty("port"));
}
```

这样我们就可以替换掉port属性，等会有写怎样去通过写入XML文件的形式读取到这个port属性！

最后是重头戏，实现Configuration类，编写ConfigurationImpl类：

按照如下形式去

```
<?xml version="1.0" encoding="UTF-8" ?>
<EMS>
    <Log class="com.briup.util.Impl.LogImpl">
    </Log>
    <gather class="com.briup.Client.Impl.GatherImpl">
        <logFile>/Users/wjh/Desktop/FirstProject/src/radwtmp</logFile>
        <positionFile>/Users/wjh/Desktop/FirstProject/src/main/resources/FilePostion.propertie
</positionFile>
    </gather>
    <EnvClient class = "com.briup.Client.Impl.EnvClientImpl">
        <host>127.0.0.1</host>
        <port>9999</port>
        <path>/Users/wjh/Desktop/FirstProject/src/BackUptmp</path>
    </EnvClient>
    <EnvServer class="com.briup.Server.Impl.EnvServerImpl">
        <port>9999</port>
    </EnvServer>
    <BackUp class="com.briup.util.Impl.BackUpImpl">
        <BackUppath>/Users/wjh/Desktop/FirstProject/src/BackUptmp</BackUppath>
    </BackUp>
    <DBStore class="com.briup.Server.Impl.DBStoreImpl">
    </DBStore>
```

```
</EMS>
```

以

```
<对象名 class="对象的全限定名">
<需要替换的属性名>属性值</需要替换的属性名>
</对象名>
```

来进行编写

在ConfigurationImpl的构造器中，编写对XML文件进行解析的代码！

这里还是用之前用到的Dom4j，先获取到根节点，然后获取最大的子节点，也就是对象名

```
fis = new FileInputStream("/Users/wjh/Desktop/FirstProject/src/main/resources/EMS.xml");
document = saxReader.read(fis);
EMS = document.getRootElement();
List<Element> EMSlist = EMS.elements();
```

使用一个map集合来存储我们得到的对象：

```
// 使用 Map 集合来存放 模块名-对象
private Map<String,WossModel> ObjectMap = new HashMap <>();
```

再遍历对象名的时候，获取对象名的class属性，也就是全限定名，通过Class.forName()来获取到的应的对象，然后存入map集合中，然后继续遍历下一层，碰到有需要可变参数的模块的时候把可变参提取出来，交给Properties，等待对象成功创建以后通过其中的init方法把Properties对象传入，最后果遇到了类继承了ConfigurationAware接口(也就是说这个类会调用其他类的对象),就调用SetConfiguation方法把Configuration对象传入！这样就可以在一个类中去调用其他类的对象了：

```
for (Element element : EMSlist){
//gather -- EnvClient -- EnvServer -- BackUp -- DBStore --
    String elementName = element.getName();
    String elementClass = element.attribute("class").getText();
    WossModel obj = (WossModel) Class.forName(elementClass).newInstance();
    if (obj instanceof ConfigurationAware){
        ((ConfigurationAware) obj).SetConfiguration(this);
    }
    // 遍历子节点，为应该赋值的变量赋值
    List<Element> ChildEMSList = element.elements();
    Properties properties =new Properties();
    for (Element element1 : ChildEMSList){
        properties.setProperty(element1.getName(),element1.getText());
    }
    // 调用其 init 方法，对变量进行赋值
    obj.init(properties);

    ObjectMap.put(elementName,obj);
}
```

当做完了这一切，对象的创建变得异常简单：

```
    @Override
    public Gather getGather() {
        return (Gather) ObjectMap.get("gather");
```

```
    }

    @Override
    public EnvClient getClient() {
        return (EnvClient) ObjectMap.get("EnvClient");
    }

    @Override
    public EnvServer getServer() {
        return (EnvServer) ObjectMap.get("EnvServer");
    }

    @Override
    public DBStore getDBStore() {
        return (DBStore) ObjectMap.get("DBStore");
    }

    @Override
    public Log getLog() {
        return (Log) ObjectMap.get("Log");
    }

    @Override
    public BackUp getBackUp() {
        return (BackUp) ObjectMap.get("BackUp");
    }
```

在配置模块之后，就可以通过这样来创造一个新的对象：

```
ConfigurationImpl configuration = new ConfigurationImpl();
Gather gather = configuration.getGather();
```

而在继承了ConfigurationAware接口的类中，可以通过

```
private Configuration configuration;
private Log logger;
@Override
public void SetConfiguration(Configuration conf) {
    this.configuration = conf;
    logger = configuration.getLog();
}
```

日志也变得如此简单：

```
logger.info("插入数据库成功：" + count + "数据");
logger.error("插入数据库失败");
```

---

项目的说明就此结束，项目的说明中的代码不一定完整，完整代码在 https://github.com/Spacider/ather-and-store

这是一个练手项目，通过这个项目你可以对java基础有更深层次的了解，其中运用了注入，模块分割常用的方法，使你之后对spring等框架的理解更深一个层次！

本人语言沟通能力尚缺，可能讲的地方会出问题，请指教！希望能和大家一起学习，一起进步！

个人网站：http://www.spacider.com/
CSDN：https://blog.csdn.net/qq_37163479
联系QQ：729215049