### 环境

可以通过在负载均衡服务器上把一个服务器的权重降为0，这样就不会访问他了，等把他设备升级好了，然后在把他的配置改回来就可以完成无缝升级了。

### 配置

xml中的这句话可以在一个xml中引入其他的xml。

注意顺序不要乱，上面的要是先引用的。

<import resource="applicationContext-system.xml"/>  
<import resource="applicationContext-data-resource.xml"/>

<context:property-placeholder ignore-unresolvable="true"

location="classpath\*:/application.properties" />

这里是classpath\*，这样的话，你的项目里可能会有很多包都有application.properties这个文件，那么他们最后都会合并到那个路径下，如果你是classpath\*的话，就会把和谐文件都加载，而如果是classpath的话，就只会加载第一个，也就是说，你classpath后面的文件在这个路径下你确定只有一个，那么就写classpath就可以了，但是对于applicaitoncontext.xml这种东西是有多个的。

<bean id="configurer"

首先这个propertyplaceholderconfiguer和上面的<context:property-placeholder>的作用是一样的，都是用来加载properties配置文件的，我们这里用的下面这个是为了根据不同的proflie加载不同系列的properties文件。  
 class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">  
 <!--<property name="properties" ref="activityConfigProperties" />-->  
 <property name="order" value="1001"/>  
 <property name="ignoreUnresolvablePlaceholders" value="true" />  
 <property name="properties" ref="configProperties"/>  
</bean>

这两个类都是propertiesfactorybean，id都是configproperties，通过beans标签的profile来区分到底用哪套环境。这都是用beans的profile来区分不同的环境的，是线上环境还是测试环境。  
<beans profile="dev">  
 <bean id="configProperties"  
 class="org.springframework.beans.factory.config.PropertiesFactoryBean">  
 <property name="locations">  
 <list>  
 <value>classpath:system-base.properties</value>  
 <value>classpath:system-dev.properties</value>  
 </list>  
 </property>  
 </bean>  
</beans>  
  
<beans profile="test">  
 <bean id="configProperties"  
 class="org.springframework.beans.factory.config.PropertiesFactoryBean">  
 <property name="locations">  
 <list>  
 <value>classpath:system-base.properties</value>  
 <value>classpath:system-test.properties</value>  
 </list>  
 </property>  
 </bean>  
</beans>

propertyPlaceholderconfiguer的order（加载顺序）和ingnoreunresloveableconfiguer的作用。

<bean id="propertyConfigurerForAnalysis" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

    <property name="location">

        <value>classpath:/spring/include/dbQuery.properties</value>

    </property>

</bean>

其中classpath是引用src目录下的文件写法。

当存在多个Properties文件时，配置就需使用locations了：(2)

<bean id="propertyConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

    <property name="locations">

       <list>

          <value>classpath:/spring/include/jdbc-parms.properties</value>

          <value>classpath:/spring/include/base-config.properties</value>

        </list>

    </property>

</bean>

 接下来我们要使用多个PropertyPlaceholderConfigurer来分散配置，达到整合多工程下的多个分散的Properties 文件，其配置如下：(3)

<bean id="propertyConfigurerForProject1" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

    <property name="order" value="1" />

    <property name="ignoreUnresolvablePlaceholders" value="true" />

    <property name="location">

       <value>classpath:/spring/include/dbQuery.properties</value>

    </property>

</bean>

 <bean id="propertyConfigurerForProject2" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

    <property name="order" value="2" />

    <property name="ignoreUnresolvablePlaceholders" value="true" />

    <property name="locations">

      <list>

        <value>classpath:/spring/include/jdbc-parms.properties</value>

        <value>classpath:/spring/include/base-config.properties</value>

      </list>

    </property>

</bean>

其中order属性代表其加载顺序，而ignoreUnresolvablePlaceholders为是否忽略不可解析的 Placeholder，如配置了多个PropertyPlaceholderConfigurer，则需设置为true

这一个true害了我两天的时间啊）

### idea

Ctrl shift r 全局搜索。

projectl路径左上角的小齿轮修改包的结构。

### 代码

@Override

使用了springboot的事务注解 @transactional 这个事务注解的value就是配置的ref了一个datasource的事务管理器。  
@Transactional(value = "accountTransactionManager")   
public int incrementMoney(int parentId, int money, String reason) {  
 if(money <= 0){  
 throw new AccountException(AccountException.COINS\_LESS\_THAN\_ZERO, "金额不能少于0");  
 }

你看由于这里是事务的，我们就可以自定义异常抛异常来终止这个函数了。  
  
 if(StringUtils.isEmpty(reason)){  
 throw new AccountException(AccountException.NEED\_REASON, "需要理由");  
 }  
  
 int oldMoney = getMoney(parentId);

数据库中查询的数据查询一次就保存下来就好了，存到一个临时变量里面。

### Sql

INSERT INTO  
 `t\_account`  
 (money, total\_money, `parent\_id`)  
 VALUES  
 (#{money}, #{money}, #{parentId})  
 ON DUPLICATE KEY UPDATE money=money+#{money},total\_money=total\_money+#{money}  
insert into xxxx values xxxx on duplicate key update xxxx

.html结尾的路径也可以？Xx&xx拼东西

### 日志

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.6.1</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>1.6.1</version>

导入这两个依赖，slf4j是用来包装日志的，而log4j是产生日志的，如果你要包装log4j，那么就要导入slf4j-log4j12包。

final Logger logger = LoggerFactory.getLogger(Test.class);

Java代码就是LoggerFactory.getLogger(xx); 获取logger对象就可以了。

public static void main(String[] args) {

logger.info("Method:Hello");

public class Test {

private static

logger.info("World");

这里logger.info啥的就可以了。

}

如果没有slf4j

if (logger.isDebugEnabled()) {

logger.debug("Processing trade with id: " + id + " symbol: " + symbol);

}

看到了么，上面是字符串的拼接。

如果有slf4j，看到了么，

ogger.debug("Processing trade with id: {} and symbol : {} ", id, symbol);

后面是占位符就和mme日志差不多。

用来slf4j的占位符是{}

### Yml

Yml方式和properties可以一起生效的，肯定是yml方式比较好咯，properties有很多的重复的咯。

### Jar包

生成jar包，mvn install 这样弄完后就在target目录生成了jar包。

Java -jar就可以执行这个jar包或者war包了，但是必须要配置入口类哦。

<https://blog.csdn.net/u011535541/article/details/75044244>

### 英文

Qps 每秒查询

Tps 每秒事务

Rt 时延

### 轻课环境

[39.107.224.104](/Users/qingclass\\x/39.107.224.104) [test.api.admin.xiwangjia.cn](http://test.api.admin.xiwangjia.cn)

[39.107.224.104](/Users/qingclass\\x/39.107.224.104) [test.admin.xiwangjia.cn](http://test.admin.xiwangjia.cn)

#[39.105.78.15](/Users/qingclass\\x/39.105.78.15) [api.admin.xiwangjia.cn](http://api.admin.xiwangjia.cn)

[101.200.56.246](/Users/qingclass\\x/101.200.56.246) [test.admin.zuul.kingofwords.cn](http://test.admin.zuul.kingofwords.cn)

[101.200.56.246](/Users/qingclass\\x/101.200.56.246) [api.test.admin.zuul.kingofwords.cn](/Users/qingclass\\x/api.test.admin.zuul.kingofwords.cn)

[101.200.56.246](/Users/qingclass\\x/101.200.56.246) [test.admin.kingofwords.cn](http://test.admin.kingofwords.cn)