参考：<http://blog.csdn.net/u012373815/article/details/53837345>

本文目录结构   
- 1.规则引擎面临的问题   
- 2. DSL领域特殊语言   
- 3. DSl 好处   
- 4. DSl 实现   
- 5. DSL －小例子   
- 6 附 转换后规则

## 1. 规则引擎面临的问题

业务规则的实现大部分是由开发人员来实现的   
业务规则需要业务分析人员能够阅读和理解   
业务规则的可读性和用户的友好性都不太好

## 2. DSL领域特殊语言

DSL == Domain Specific Language   
以贴近业务领域的方式,即以类**[自然语言](http://lib.csdn.net/base/nlp" \t "_blank" \o "自然语言理解和处理知识库)**的方式来构造软件,使得我们不用花费太多精力就能看懂代码所对应的业务含义   
它是创建规则语言的一种方式，致力于解决我们的问题域   
DSL相当于一个转换器,它能将某一领域内的术语转换成规则语言

## 3. DSL好处

领域专家只需要业务，而不需要关注技术   
规则模式可以被重复利用   
使业务领域专家更平滑的将业务领域中的概念转换成DSL（存放于dslr文件中）   
提高业务规则的可读性

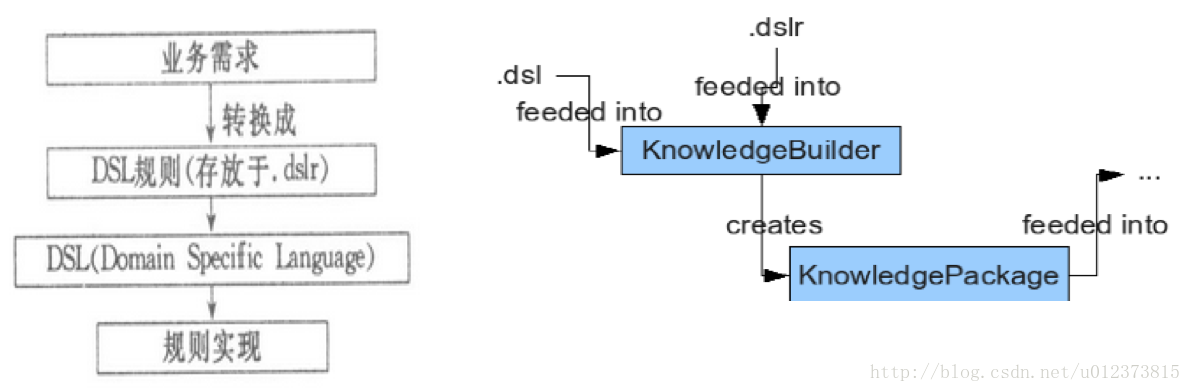
## 4. DSL 实现

### 4.1 DSL包含两种类型的文件

dslr 文件：存放了一些业务领域专用语言   
dsl文件：定义了领域专用语言到规则语言的映射，能将领域专用语言转换成规则语言

### 4.2 转换过程

dslr文件通过dsl转换之后的结果是一个符合语法的DRL文件（自然语言到规则语言的转换）



## 5. DSL －小例子

用户买电影票小例子。   
Silver卡用户, 买票获得10元代金劵；   
Gold卡用户, 买票获得30元代金劵；   
Platinum卡用户, 买票获得50元代金劵；

### 5.1POJO

有两个 pojo 对象 一个是用户，一个是电影票

Customer

package com.us.ticketDSL;

/\*\*

\* Created by yangyibo on 16/12/21.

\*/

public class Customer {

private String name;

private String subscription;

public Customer() {

}

public Customer(final String name, final String subscription) {

super();

this.name = name;

this.subscription = subscription;

}

public String getName() {

return this.name;

}

public String getSubscription() {

return this.subscription;

}

public String toString() {

return "[Customer " + this.name + " : " + this.subscription + "]";

}

}

Ticket

package com.us.ticketDSL;

/\*\*

\* Created by yangyibo on 16/12/21.

\*/

public class Ticket {

private Customer customer;

private String status;

public Ticket() {

}

public Ticket(final Customer customer) {

super();

this.customer = customer;

this.status = "买票";

}

public String getStatus() {

return this.status;

}

public void setStatus(final String status) {

this.status = status;

}

public Customer getCustomer() {

return this.customer;

}

public String toString() {

return "[好看电影院: 尊敬的"+this.customer.getSubscription()+" 卡用户 " + this.customer.getName() + ",您" + this.status + "]";

}

}

5.2 kmodule.xml

如果没有 kmodule.xml 文件，在META-INF 文件夹下新建kmodule.xml 文件填写如下内容。

<?xml version="1.0" encoding="UTF-8"?>

<kmodule xmlns="http://jboss.org/kie/6.0.0/kmodule">

<!--DSL买票-->

<kbase name="TicketWithDSLKB" packages="com.us.ticketDSL">

<ksession name="TicketWithDSLKS"/>

</kbase>

</kmodule>

5.3 DSL 文件

注意：DSL文件和DSRL 文件之间的关系是条件对应替换，例如我DSRL 文件中的“ 我要买张票 "买票"“ 通过DSR 文件转换，转换后为

“ customer : Customer( )

ticket : Ticket( customer == customer, status == "{status}" )“

本文的DSL和DSLR文件替换后如文章末尾“

附 转换后规则 “展示所示。

ticket.dsl文件内容

[condition][]我要买张票 "{status}"=customer : Customer( ) ticket : Ticket( customer == customer, status == "{status}" )

[condition][]这是一个 "{subscription}" 卡客户 "{status}"=customer : Customer(subscription == "{subscription}") ticket : Ticket( customer == customer, status == "{status}")

[consequence][]打印 "{message}"=System.out.println(customer.getName()+"{message} ");

[consequence][]优惠=ticket.setStatus("获得了一张代金劵");update(ticket);

[consequence][]Send email=sendEscalationEmail( customer, ticket );

5.4 DSLR 文件

ticketing.dslr 文件内容

注意：duration 1000 属性是指此规则匹配成功后，另开线程1秒后执行，

package com.us.ticketDSL

import com.us.ticketDSL.Customer;

import com.us.ticketDSL.Ticket;

expander ticketing.dsl

rule "New Ticket"

salience 10

when

我要买张票 "买票"

then

打印 " 买到了一张票"

end

rule "Silver "

duration 3000

when

这是一个 "Silver" 卡客户 "买票"

then

优惠

打印 " 是个Silver卡用户, 买票获得10元代金劵。"

end

rule "Gold "

duration 1000

when

这是一个 "Gold" 卡客户 "买票"

then

优惠

打印 " 是个Gold卡用户, 买票获得30元代金劵。"

end

rule "Platinum "

when

这是一个 "Platinum" 卡客户 "买票"

then

优惠

打印 " 是个Platinum卡用户, 买票获得50元代金劵。"

end

rule "Escalate"

when

我要买张票 "获得了一张代金劵"

then

Send email

end

function void sendEscalationEmail( Customer customer, Ticket ticket ) {

System.out.println( "Email : " + ticket );

}

5.5 测试

package com.us.ticketDSL;

import org.kie.api.KieServices;

import org.kie.api.runtime.KieContainer;

import org.kie.api.runtime.KieSession;

/\*\*

\* Created by yangyibo on 16/12/21.

\*/

public class TicketExampleWithDSL {

public static void main(final String[] args) {

KieContainer kc = KieServices.Factory.get().getKieClasspathContainer();

execute( kc );

}

public static void execute( KieContainer kc ) {

KieSession ksession = kc.newKieSession( "TicketWithDSLKS" );

final Customer a = new Customer( "刘德华",

"Gold" );

final Customer b = new Customer( "郭富城",

"Platinum" );

final Customer c = new Customer( "张学友",

"Silver" );

final Customer d = new Customer( "黎明",

"Silver" );

final Ticket t1 = new Ticket( a );

final Ticket t2 = new Ticket( b );

final Ticket t3 = new Ticket( c );

final Ticket t4 = new Ticket( d );

ksession.insert( a );

ksession.insert( b );

ksession.insert( c );

ksession.insert( d );

ksession.insert( t1 );

ksession.insert( t2 );

ksession.insert( t3 );

ksession.insert( t4 );

ksession.fireAllRules();

try {

System.err.println( "[[ Sleeping 5 seconds ]]" );

Thread.sleep( 5000 );

} catch ( final InterruptedException e ) {

e.printStackTrace();

}

System.err.println( "[[ awake ]]" );

ksession.fireAllRules();

ksession.dispose();

}

}

输出：   
刘德华 买到了一张票   
郭富城 买到了一张票   
张学友 买到了一张票   
黎明 买到了一张票   
郭富城 是个Platinum卡用户, 买票获得50元代金劵。   
[[ Sleeping 5 seconds ]]   
Email : [好看电影院: 尊敬的Platinum 卡用户 郭富城,您获得了一张代金劵]   
[[ awake ]]   
张学友 是个Silver卡用户, 买票获得10元代金劵。   
黎明 是个Silver卡用户, 买票获得10元代金劵。   
刘德华 是个Gold卡用户, 买票获得30元代金劵。   
Email : [好看电影院: 尊敬的Silver 卡用户 张学友,您获得了一张代金劵]   
Email : [好看电影院: 尊敬的Silver 卡用户 黎明,您获得了一张代金劵]   
Email : [好看电影院: 尊敬的Gold 卡用户 刘德华,您获得了一张代金劵]

6 附 转换后规则，只为加深理解

实际上DSL 在执行是会在内部进行替换，按照本文DSL 和DSLR的替换结果应该如下：

注意：（此处代码只是用于理解DSL和DSLR 的替换，不用写在代码中）

package com.us.ticketDSL

import com.us.ticketDSL.Customer;

import com.us.ticketDSL.Ticket;

function void sendEscalationEmail( Customer customer, Ticket ticket ) {

System.out.println( "Email : " + ticket );

}

rule "New Ticket"

salience 10

when

customer : Customer( )

ticket : Ticket( customer == customer, status == "买票" )

then

System.out.println(customer.getName() + "买到了一张票" );

end

rule "Silver "

duration 3000

when

customer : Customer( subscription == "Silver" )

ticket : Ticket( customer == customer, status == "买票" )

then

ticket.setStatus("获得了一张代金劵");update(ticket);

System.out.println(customer.getName() + "是个Silver卡用户, 买票获得10元代金劵。" );

end

rule "Gold "

duration 1000

when

customer : Customer( subscription == "Gold" )

ticket : Ticket( customer == customer, status == "买票" )

then

ticket.setStatus("获得了一张代金劵");update(ticket);

System.out.println(customer.getName() + "是个Gold卡用户, 买票获得30元代金劵。" );

end

rule "Platinum "

when

customer : Customer( subscription == "Platinum" )

ticket : Ticket( customer == customer, status == "买票" )

then

ticket.setStatus("获得了一张代金劵");update(ticket);

System.out.println(customer.getName() + "是个Platinum卡用户, 买票获得50元代金劵。" );

end

rule "Escalate"

when

customer : Customer( )

ticket : Ticket( customer == customer, status == "获得了一张代金劵" )

then

sendEscalationEmail( customer, ticket );

end