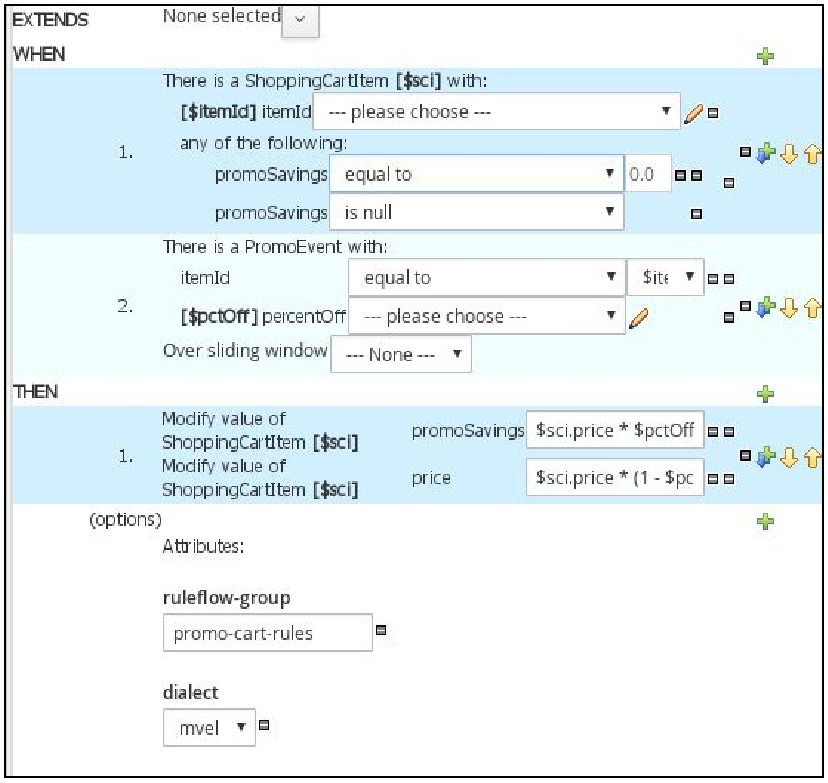
Event Processing in Business-Central Demo

In this demo we are going to demonstrate some of the capabilities of Complex Event Processing with BRMS through business-central. To work on this demo you can edit master or make a branch off of master in business-central. The finished product of the demo is in the finishedDemo branch.

Project Use Case and Overview:

This demo project implements an online store. We will be using BRMS CEP capabilities to process events called ‘PromoEvents’. Currently our store rules assume that there will be at most one promotion available per item in your shopping cart. This logic is implemented with this Guided Rule in business-central:



```
EXTENDS   None selected
WHEN
1.  There is a ShoppingCartItem [$sci] with:
      [$itemId] itemId   --- please choose ---
      any of the following:
        promoSavings   equal to   0.0
        promoSavings   is null
2.  There is a PromoEvent with:
      itemId   equal to   $ite
      [$pctOff] percentOff   --- please choose ---
    Over sliding window   --- None ---
THEN
1.  Modify value of ShoppingCartItem [$sci]   promoSavings   $sci.price * $pctOff
    Modify value of ShoppingCartItem [$sci]   price          $sci.price * (1 - $pc
(options)
    Attributes:
    ruleflow-group
      promo-cart-rules
    dialect
      mvel
```

This rule can be found in the master branch of the project in src/main/resources/Apply Cart Item Promotions.rdr

In the rule we are not using any temporal event attributes. We simply only care if there is a *PromoEvent* that has the same ‘itemId’ as an item in a cart. If this is true we apply the ‘percentOff’ to the *ShoppingCartItem* ‘price’ and add the correct value to the ‘promoSavings’.

New Business Requirements:

Lets assume that the business has come to us and told us that it is possible that there may be more than one PromoEvent for a single ShoppingCartItem. This means we will have to edit our current rule and do some more complex event processing. We are given this specific business scenario:

Given a ShoppingCartItem with no promo savings added
And multiple PromoEvents for that ShoppingCartItem
When the “Apply Cart Item Promotions” rule is fired
Then the newest PromoEvent should be used for the ShoppingCartItem

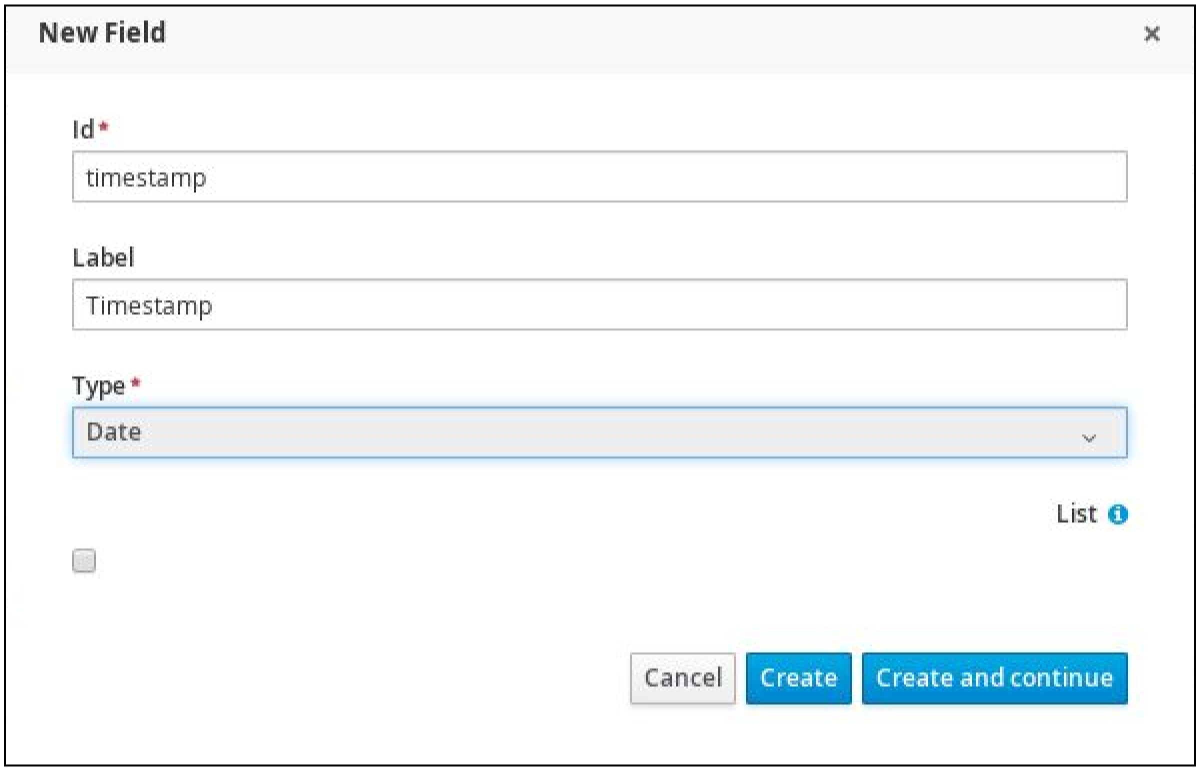
Editing the Event Data Object

First we want to add a ‘Timestamp’ attribute to our PromoEvent class and map it to a data field. If you do not do this the PromoEvent Timestamp will be set to the current system time when it is inserted into the kie session. This is fine to do, but for this demo we are assuming that want to use the a time from some external system where we are retrieving our PromoEvents from as our event timestamp.

1. Navigate to src/main/java/com/redhat/coolstore/PromoEvent.java.

2. In the Editor tab click the “add field” button

3. Fill out the form as follows then click “Create” :


New Field

Id*
timestamp

Label
Timestamp

Type*
Date

List

4. Open the Source tab and add this above the class definition:

   @org.kie.api.definition.type.Timestamp("timestamp")

5. Click the “Save” Button in the top right hand corner

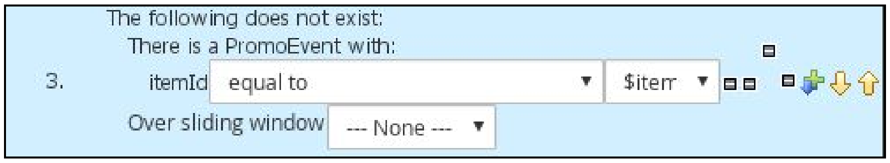
Edit the Rule

Now we can edit the rule to add in our new logic. We will be using the “after” operator to find a PromoEvent for a certain ShoppingCartItem that does NOT have any PromoEvents for the same item inserted “after” it. In other words we want to use the most recently added PromoEvent for every ShoppingCartItem promotion.

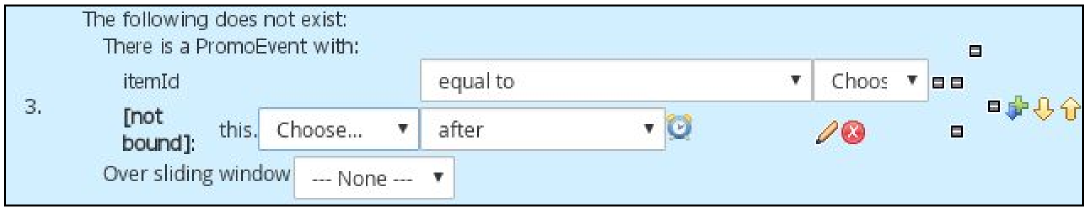
1. Navigate to src/main/resources and open the Apply Cart Item Promotions.rdrl file
2. Click the plus button next to the “WHEN” header
3. Choose “The following does not exist …” from the menu. In the top left corner make sure the Position drop down menu has “Bottom” chosen. Click the “Ok” button. This will add a new when condition at the bottom.
4. ***If adding this condition causes all but your first condition to disappear (this is a bug in BC) try choosing “Top” in the Position drop down menu when adding the condition. The

rest of the conditions will still disappear but once you follow step 5 they will all reappear and you can move the condition down to the bottom with the yellow down arrow on the side: ***

5. You will see a red note in the condition that says “click to add patterns”. Click this and you will see a pop up window
6. In the pop up window click on the “choose fact type” drop down and choose the “PromoEvent”. This will add a “There is a Promo Event” statement to your condition.
7. Click on that new statement and click on the “Add a restriction on a field” drop down in the pop up window. Choose “itemId”.
8. Click on the new drop down menu in your condition and choose “equal to”.
9. Click on the pencil icon next to the drop down menu
10. In the pop up window click the “Bound Variable” button. A new drop down menu will appear where the pencil icon was
11. In the new drop down menu choose “$itemId”. This finishes the first part of our condition statement which is checking that there is NOT another PromoEvent with the same itemId as the ShoppingCartItem in our first condition.

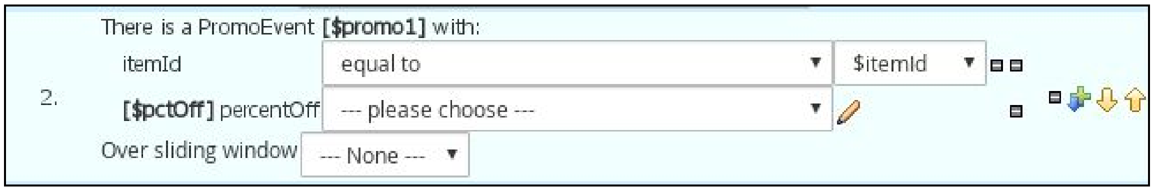

The following does not exist:
There is a PromoEvent with:
3. itemId equal to $iterr
Over sliding window --- None ---

12. Click on the “There is PromoEvent with” text to add the next part of the condition
13. Click the “Expression Editor” button in the pop up window. You will see two drop down menus in your condition
14. In the first drop down menu choose “this”. This will add another drop down menu next to it. Just ignore this drop down menu.
15. In the second drop down menu choose “after”. A clock icon will appear next to the drop down menu. This allows you to set a time range in the after condition but we do not want to do this in this demo so ignore this button.

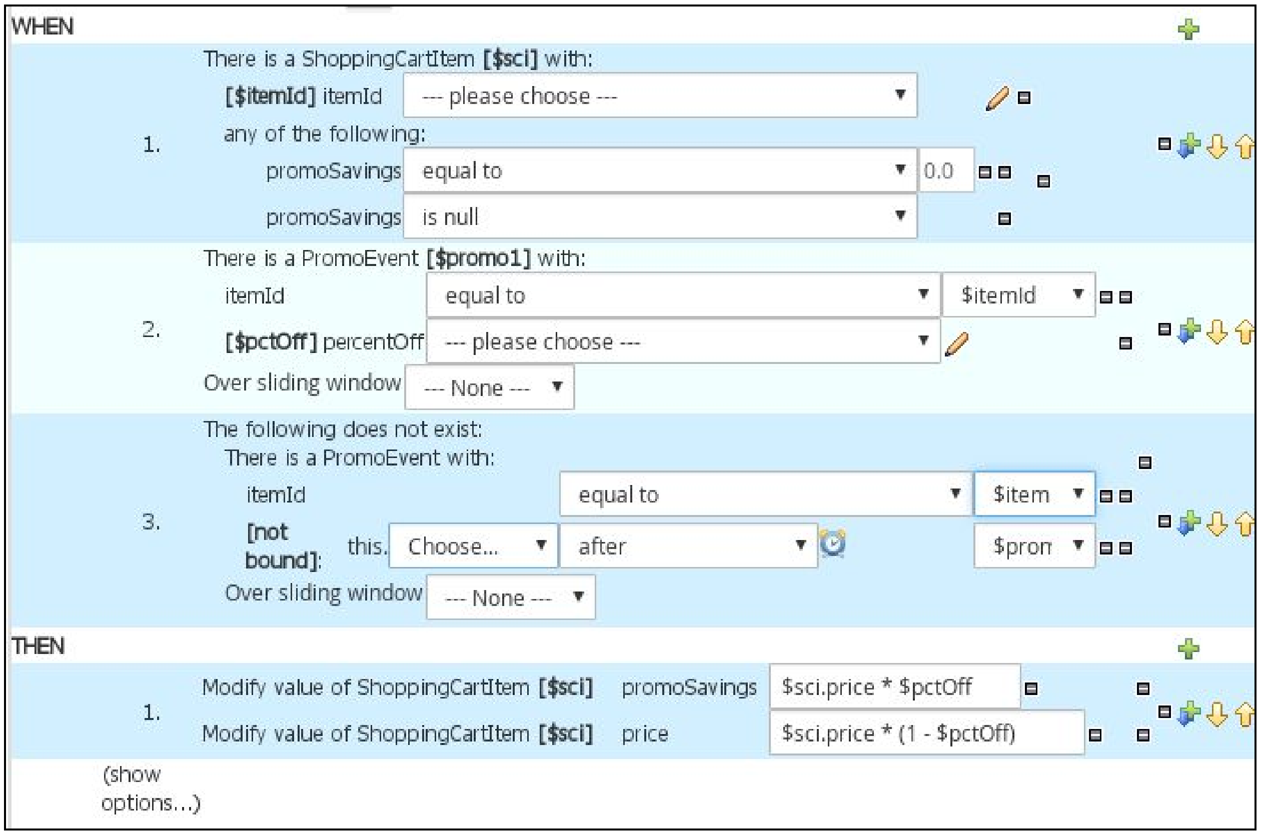

The following does not exist:
There is a PromoEvent with:
3. itemId equal to Choos
[not bound]: this. Choose... after
Over sliding window --- None ---

16. We must now bound a variable the PromoEvent in our second condition so we can use it on our new condition. Click the “There is a Promo Event with:” statement in your second condition

17. In the pop up window write “$promo1” in the “Variable name” text box and click the “Set” button. You will now see this bound variable in the second condition statement.


2. There is a PromoEvent **[$promo1]** with:
itemId | equal to | $itemId
[$pctOff] percentOff | --- please choose ---
Over sliding window | --- None ---

18. In the 3rd condition click on the pencil icon next to the clock icon
19. In the pop up window click the “Bound Variable” button. A new drop down menu will appear next to the clock icon
20. Choose “$promo1” in the new drop down menu
21. You are finished making edits to the rule! It should look like this now:


WHEN
1. There is a ShoppingCartItem **[$sci]** with:
[$itemId] itemId | --- please choose ---
any of the following:
promoSavings | equal to | 0.0
promoSavings | is null
2. There is a PromoEvent **[$promo1]** with:
itemId | equal to | $itemId
[$pctOff] percentOff | --- please choose ---
Over sliding window | --- None ---
3. The following does not exist:
There is a PromoEvent with:
itemId | equal to | $item
[not bound]: this. | Choose... | after | $pron
Over sliding window | --- None ---
THEN
1. Modify value of ShoppingCartItem **[$sci]** | promoSavings | $sci.price * $pctOff
Modify value of ShoppingCartItem **[$sci]** | price | $sci.price * (1 - $pctOff)
(show options...)

Edit the Test for the Rule:

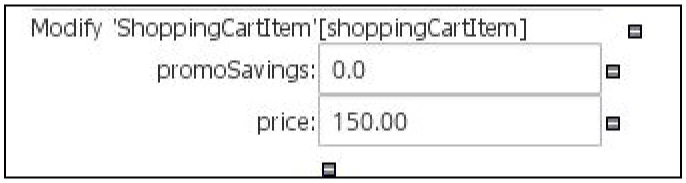
Now that we have changed the rule to fit our new business requirements we should test these new requirements. You should not rely only on business-central tests when using event processing in your rules. All rules you write in business-central should be tested outside business central using junit/cucumber or other testing frameworks. Business-central tests do not allow you to advance pseudo session clock times or even give anything more exact than the month/day/year for a timestamp. The test in this demo is just to show you that your rule runs, it does not fully test the rule functionality.

If you navigate to src/test/resources and open the “Apply Promotions Test.scenario” file you will see the test. We must make some changes to this test so it will run since we added a new field to the PromoEvent

1. In the GIVEN statement click the “Insert ‘PromoEvent’” statement
2. In the pop up window choose timestamp from the drop down menu and click the “Ok” button
3. Click on the blank field next to “timestamp” and choose June 28th 2017 from the calendar

If you click the “run scenario” button in the top right hand corner you will see that this test now passes after adding in the new timestamp field. This is because we have only one PromoEvent inserted into the session, therefore the rule conditions are all still true. This does not test a scenario where we have multiple PromoEvents for a single ShoppingCartItem so lets add to the test to cover that.

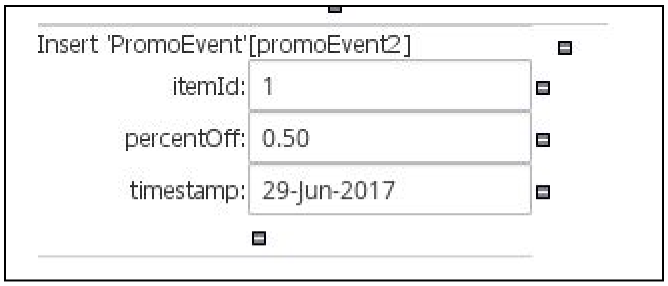
1. In the Apply Promotions Test.scenario file click the “More ..” button below the “EXPECT” statements. This will add another GIVEN, WHEN, THEN below the previous test
2. Click the plus sign next to the GIVEN
3. In the pop up window in the “Modify an existing fact” drop down menu choose “shoppingCartItem” and click Add
4. Click the “Add a field” statement in the new statements that have just appeared in your GIVEN
5. In the pop up window choose “promoSavings” and click “Ok”
6. Click the pencil icon that has just appeared next to “promoSavings” in your GIVEN
7. In the pop up window click the “Literal Value” button
8. The pencil icon is now replaced with a text box. In this box type “0.0”
9. Click the “Modify ‘ShoppingCartItem’ “ statement to add a new field to modify
10. In the pop up window choose “price” and click “Ok”
11. Click the pencil icon that has just appeared next to “price” in your GIVEN
12. In the pop up window click the “Literal Value” button
13. The pencil icon is now replaced with a text box. In this box type “150.00”. You have just finished the first half of you GIVEN statement. This first half modifies the ShoppingCartItem price and promoSavings so that the “Apply Cart Item Promotions” rule



Modify 'ShoppingCartItem'[shoppingCartItem]

promoSavings:	0.0
price:	150.00

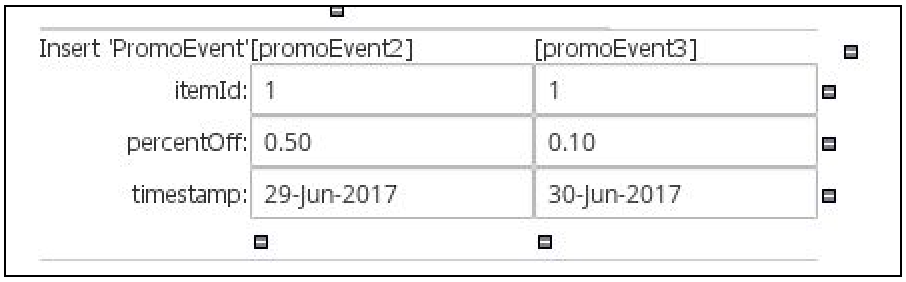
14. will pick up the ShoppingCartItem again
15. Click the plus sign next to the GIVEN again
16. In the pop up window choose PromoEvent from the “Insert a new fact” drop down menu. Type “promoEvent2” in the “Fact name” text box and click the “Add” button
17. You will see a new statement appear “Insert ‘PromoEvent’ “. The fields for PromoEvent are below with pencil icons next to them. Click on the pencil icons and click the “Literal Value” button in the pop up window for each field.
18. Fill out the text fields as follows:



Insert 'PromoEvent'[promoEvent2]

itemId:	1
percentOff:	0.50
timestamp:	29-Jun-2017

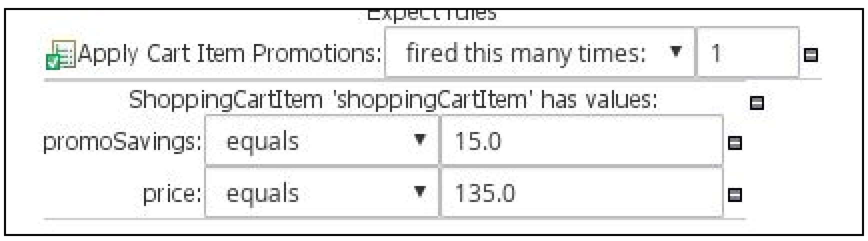
19. Click the plus sign next to the GIVEN again
20. In the pop up window choose PromoEvent from the “Insert a new fact” drop down menu. Type “promoEvent3” in the “Fact name” text box and click the “Add” button
21. You will see “promoEvent3” will pencil signs for each field now next to “promoEvent2” fields. Click on the pencil icons and click the “Literal Value” button in the pop up window for each field.
22. Fill out the text fields as follows:



Insert 'PromoEvent'	[promoEvent2]	[promoEvent3]
itemId:	1	1
percentOff:	0.50	0.10
timestamp:	29-Jun-2017	30-Jun-2017

23. Click the plus sign next to the EXPECT
24. In the pop up window, type “Apply Cart Item Promotions” in the “Rule” text field and click the “OK” button

25. In the new drop down menu next to “Apply Cart Item Promotions:” choose “fired this many times”
26. In the new text field next to the drop down menu type “1”
27. Click the plus sign next to the EXPECT again
28. In the drop down menu, in the “Fact value:” drop down menu choose shoppingCartItem. Click the Add button.
29. Click the new statement “ShoppingCartItem ‘shoppingCartItem’ has values:”
30. In the pop up window choose “promoSavings”
31. Click the statement again and in pop up menu choose “price”
32. Fill the value out as shown:



Expect rules

Apply Cart Item Promotions: fired this many times: 1

ShoppingCartItem 'shoppingCartItem' has values:

promoSavings: equals 15.0

price: equals 135.0

33. Click the “Save” button in the top right hand corner and then the “Run scenario” button. The test should pass.