IBM OpenPages with Watson
Version 8.2.0

Trigger Developer Guide

Copyright Information

Licensed Materials - Property of IBM Corporation.

© Copyright IBM Corporation, 2003, 2020.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written.

These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Open Source Software Listing

A listing of all open source software used in the IBM® OpenPages® with Watson™ product can be found in the `<OP_Home>|Notices|Notices.doc` file, located in the root folder of your OpenPages installation (where `<OP_Home>` by default is `c:\OpenPages` or `/opt/OpenPages`).

Trademark Information

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following terms are trademarks or registered trademarks of other companies:

- Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Disclaimer

Although every precaution has been taken in the preparation of this document, IBM assumes no responsibility for errors or omissions. No liability is assumed for damages resulting from the use of the information contained herein.

Contents

Introduction 7
Audience 7
Finding information 7
Trigger definition 7
High-level Features 8
GRC Trigger Framework 8
Position 8
Transactions 9
Flow Diagram 9
Configuring Triggers 9
GRCTrigger Properties 11
Rule Properties 11
Rule Attributes 12
Event Handler Properties 12
Event Handler Attributes 12
Supported Events 12
Create Object 12
Create Object with Relations 12
Update Object 13
Associate Objects 13
Disassociate Objects 13
Delete Objects 13
Reassign Primary Parent Object 13
Move Object 14
Move Objects 14
Rename Object 14
Rename Objects 14
Copy Object 14
Copy Objects 15
Search Objects 15
Search Objects 15
Configure Event Handlers to Execute Serially 16
Implementing a Rule 16
Implementing an Event Handler 17
Out-of-the-Box Rules 17
Content Type Match Rule 17
Detect Property Change Rule 18
Fields Match Rule 19
Folder Match Rule 20
Out-of-the-box Handlers 22
Date Validation Handler 22
Send Email Event Handler 22
Set Current Date Handler 24
Set Enum Field Handler 24
Trigger Events 24
Disabling Triggers 25
Error Handling 25
Configuring Lifecycle Triggers 27
Lifecycle trigger Properties 27
Lifecycle trigger Email Properties 28
Lifecycle trigger transition element and Properties 28
Lifecycle trigger defaultsettings element and Properties 29

Lifecycle trigger setfield element and Properties 30
Lifecycle trigger conditions element and Properties 31
Lifecycle trigger condition element and Properties 32
Samples 35
Abstract Resource Trigger 36
Folder Match Trigger 37
Detect Property Change Trigger 37
Content Type Match Trigger 37
Legacy Out-of-the-Box Actions 38
Abstract Resource Trigger Action 38
Abstract Change Control Trigger Action 38
Change Control Flag Trigger Action 38
Abstract Exchange Rate As Of Date Trigger Action 39
Abstract Picklist Dependency Trigger Action 39
Picklist Value Mapping Trigger Action 40
Abstract Resource Based Send Email Action 41
Default Send Email To User In Field Action 41
Send Email To User In Field Action 42
Set Field Value From Another Field Action 43
Set Field Value From Parent Action 43
Abstract Resource Copy Trigger Action 44
Best Practices 45
Tips 46
Legacy Trigger Rules and Actions 47
Rule Properties 47
Rule Attributes 48
Action Properties 48
Action Attributes 48

Introduction

The IBM® OpenPages® GRC Trigger Framework provides the infrastructure to support the implementation and deployment of custom business logic and rules. This framework is built in Java and allows the implementation of any business logic based on existing functionality available through the OpenPages GRC API, which can be invoked when users perform an action in the system. These business rules can be leveraged to enable more automation and validation in IBM® OpenPages® with Watson™ as users use the application.

Audience

To use the Trigger Developer Guide effectively, you should be familiar with the following:

- OpenPages with Watson application and usage
- OpenPages GRC API
- Programming languages and integrated development environments (IDEs), such as the Java™ programming language and the Eclipse IDE

Finding information

To find IBM OpenPages with Watson product documentation on the web, including all translated documentation, access the IBM OpenPages with Watson Knowledge Center (http://www.ibm.com/support/knowledgecenter). Release Notes are published directly to the Knowledge Center and include links to the latest technotes and APARs.

Trigger definition

A trigger is a piece of code that can be added before or after the execution of an operation is performed on the OpenPages platform. This piece of code can perform anything that is written in Java.

A trigger consists of the following two parts:

- A rule - this is a condition that applies to the operation being executed and the parameters involved in the operation. For example,
  - The operation being executed
  - Type of object
  - Condition on a property of the object(s) in context

- One or more event handlers - an event handler is executed if the current operation satisfies the rule defined for the trigger. These actions can perform any business logic. For example,
  - Throw a validation error
  - Create a new object
  - Delete an existing object
  - Reset or modify properties of an object
  - Modify properties of a related object
  - Execute a report or program
  - Kick off a workflow

High-level Features

Triggers have the following characteristics:

- Are only available for specific platform operations.
- Triggers must be written in Java.
- Can invoke any program or module that can be called from Java.
- Can be used with all functionality that is supported by the OpenPages GRC Platform API.
- Have access to the current user's OpenPages session.
- Are executed within the existing transaction boundary of the original operation.
- Can be configured to execute before or after the original operation execution.
- When a user performs an operation that may have triggers, the framework determines which triggers are applicable and invokes them.

GRC Trigger Framework

The GRC Trigger framework performs the execution of triggers in the OpenPages system. The framework handles loading of custom code through dynamic class loading during OpenPages start up. The definition of triggers is provided in an XML configuration file that is stored in the OpenPages repository and also loaded during start up. The XML configuration file will provide the class names to be invoked and the necessary properties and attributes.

As operations take place in the system, either from a user or system automation, events are generated in the trigger framework by a subset of supported actions. As these events are generated, any triggers that are registered through configuration for different event types have the opportunity to handle these events and perform additional operations. For example, when a user creates a new GRC Object in the OpenPages user interface, a CREATE_OBJECT event is generated and triggers may be configured to handle CREATE_OBJECT events to perform additional automation on the user's behalf.

The custom classes specified in the trigger XML configuration need to be in the classpath. The options to have them in the classpath are:

- Add them to the openpages-ext.jar, which is located in <InstallDir>/aurora/lib.
- Create a new jar file with the custom classes and add the jar to the <InstallDir>/aurora/op-ext-lib directory

Note: After you make changes to a class file and add it to a jar file, you must restart OpenPages to utilize the modifications in the class.

Position

Events are generated in one of two phases of an operation, PRE or POST. Triggers are registered to listen for either one or other position.

- **PRE** - are events that happen prior to the operation actually being performed by the system. For example, during the creation of a GRC Object, a PRE event has all the information about the object to be created, but the system has yet to take action to create the object and persist values.
- **POST** - are events that happen after the operation has been performed by the system and before the transaction has been committed; allowing for further processing of additional business logic.

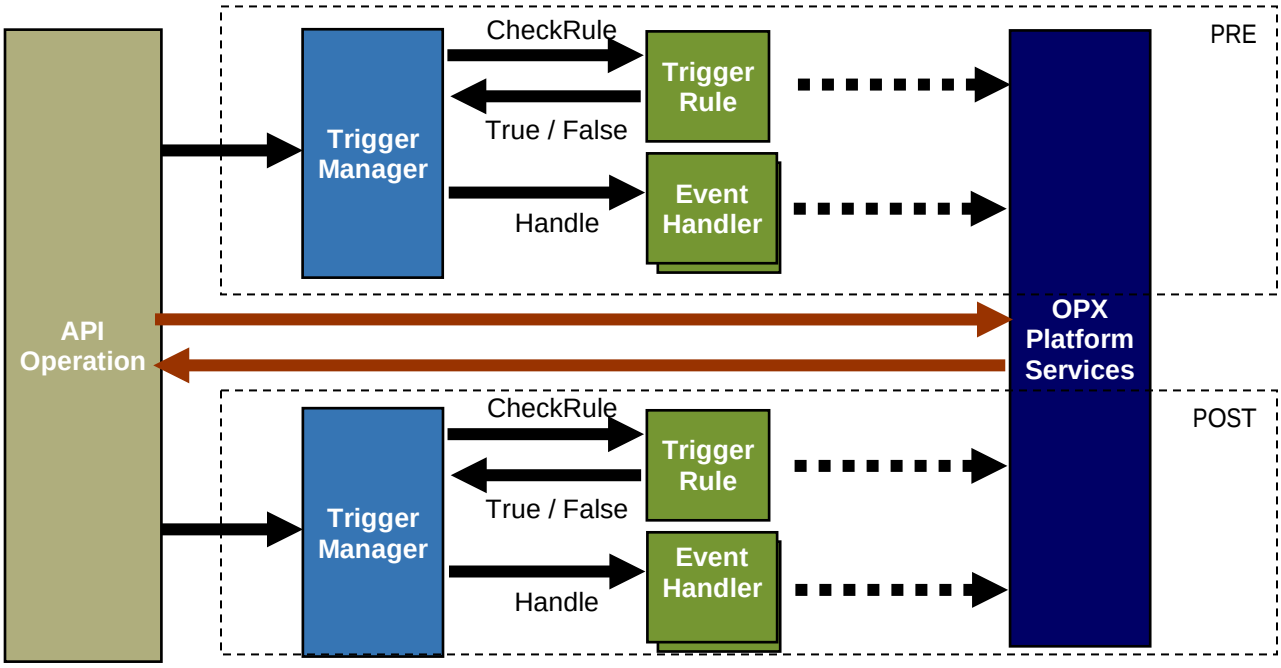
The position may affect the availability of certain information and methods within the trigger context for the rules and event handlers. Please refer to the individual event types for more detail.

Transactions

All triggers on operations are executed within the same transaction of the original system operation. If an error occurs in a trigger, whether system or business logic, the framework will roll back the transaction. In other words, the original operation will be rolled back if any error occurs.

Flow Diagram

The following diagram shows trigger interactions with an API operation. The API operation represents the call made by the application when an action is taken by a user or the system that calls one of the specified APIs that support triggers. The Trigger Manager is the component of the GRC Trigger Framework that translates the API operation into an event, checks for any registered triggers, and invokes their methods. This results in calls that interact with the platform to perform the automation business rules.



Configuring Triggers

Triggers are configured in XML documents. The documents are saved as resource objects in the OpenPages repository as system files. In previous releases, system files were managed in the OPX administrative interface. Starting with OpenPages® 8.0.0, these functions are moving to the primary administration user interface.

One of the configuration files named '_trigger_config_.xml' is located under ***Administration > Manage System Files -> SysXMLDocument*** . There is also a folder named ***'TriggerConfigFiles'*** that is located under ***SysXMLDocument***. This folder can contain additional trigger configuration files.

You can check-out and download a configuration file to modify it and perform a check-in when finished with any changes.

Note: For details see the “System file Management” chapter in the IBM® OpenPages® GRC Platform Administrator’s Guide.

To load a trigger configuration file at startup, the configuration file name must be added to a registry setting.

To add a file to the configuration registry setting:

1. Log in to the OpenPages application user interface as a user with administrative privileges for Settings.
2. Go to Administration > Settings menu
3. Expand the Settings tree to find the trigger configuration setting (Applications | GRCM | Trigger Configuration Files)
4. Edit the setting to add one or more of the trigger configuration files that were added as system files. For Example:
   _trigger_config_.xml,openpages-solutions.xml,OPLC-QuestionnaireAssessment.xml,OPLC-Incident.xml

During startup, the files listed in the “Trigger Configuration Files” registry setting are loaded. The files are loaded in the order that they are listed in the setting.

Important: Starting with 8.0.0.1, any modification made to the Trigger configurations can be loaded by running the “Trigger Configuration Refresh Utility” under Reporting -> OpenPages V6->Configuration Utilities.

Running the report refreshes the trigger cache and displays the message 'Trigger Configuration Refreshed.' when complete.

Access to this menu is controlled via the OPX administrative interface. By default, only users in the OPAdministrators group and the OpenPagesAdministrator users have access to this utility

Updates to JAR files are not refreshed by this utility. Any modification made to the JAR files will only be loaded during the next server startup.

Prior to 8.0.0.1, Trigger configurations were loaded during server startup only. Any modification made to the configurations were loaded during the next server startup.

The structure of the _trigger_config_.xml document is:

```
<trigger-definitions>
  <grcTrigger name="Name of the trigger 1"
           event="name.of.event"
           position="PRE or POST">
    <rule class="classname.for.rule" >
        <attribute name="rule.attribute1" value="value"/>
        <attribute name="rule.attribute2" value="value"/>
    </rule>
    <eventHandler class="classname.for.eventhandler">
        <attribute name="action.attribute1" value="value"/>
    </eventHandler >
  </grcTrigger>
</trigger-definitions>
```

To define a rule with event handlers in the trigger framework, add a new <grcTrigger> element with one <rule> and multiple <eventHandler> in the XML configuration file. The <attribute> tags are determined by

the rule or event handler configuration requirements and are used to pass configurable parameters to the rule or event handler classes. The rule and eventHandler elements will refer to the classes with the "class" attributes.

GRCTrigger Properties

The <grcTrigger> element defines the trigger and contains the rule and event handlers.

name
This represents the name of the GRC Trigger and must be unique across all triggers configured in the system.

event	
The operation to which this trigger will be applied.	
The possible values are:	
create.object	Creates an object
create.object.with.relations	Creates an object with related parent/child associations
update.object	Updates an object
associate.objects	Associates one or more objects
disassociate.objects	Disassociates one or more objects
delete.objects	Deletes a tree of objects
reassign.primary.parent	Reassigns primary parent object
move.object	Moves an object from one location to another
move.objects	Moves a tree of objects
rename.object	Rename an object
rename.objects	Rename a self-contained object
copy.object	Copies an object from one location to another
copy.objects	Copies a tree of objects
search.objects	Legacy searches performed by the OpenPages platform using the OPSDK
query.objects	Queries executed by the API

position	
This defines the position where the trigger should be executed, whether before or after the operation completes execution.	
The possible values are:	
PRE	Before execution
POST	After execution

Rule Properties

The <rule> element configures which class will be used to see if the event should be handled by the trigger. Attributes are used to configure the behavior of the rule.

classname
This defines the class name where the business logic of the rule is implemented. Note that this class has to exist in the runtime class path.

Rule Attributes

One or more custom attributes can be defined for rules. These attributes are simple name value pairs and are specific to the rule implementation.

Event Handler Properties

classname
This defines the class name where the business logic of the event handler is implemented. Note that this class has to exist in the runtime class path.

exclusiveOperationBy
This is an optional property to control the concurrent operation of triggers. Refer to the **Configure Event Handlers to Execute Serially** section for more details.

Event Handler Attributes

One or more custom attributes can be defined for these event handlers. These attributes are simple name value pairs and are specific to the event handler implementation.

Supported Events

The following operations are supported. For details on methods available on the events, refer to the IBM OpenPages GRC API Javadocs.

Create Object

Syntax: create.object

Description: Creates a resource object.

Availability during this operation: The object that is being created is available before and after the operation. In the POST phase, the object that was created is available.

Create Object with Relations

Syntax: create.object.with.relations

Description: Creates a resource object with related parent/child associations. This is a new event type since OpenPages 8.2. Previously, when a resource object is created via OpenPages user interface or via API, multiple trigger events are fired – one for the create.object and zero to a few for associate.objects. It was not easy to validate all the non-primary parent associations or child associations that happens on the creation view. With the new create.object.with.relations event addition, the event handler can only get this event once for PRE trigger and another for POST trigger. For the backward compatibility, the system still fires create.object event and a series of associate.objects events. The trigger implementation code can skip the old events by checking AbstractEvent#hasOuterCreateWithRelationsEvent() flag.

Availability during this operation: The object that is being created and its parent/child associations are available before and after the operation. In the POST phase, the object that was created is available.

Update Object

Syntax: `update.object`

Description: Updates a resource object.

Availability during this operation: The object being updated is available before and after the operation.

Associate Objects

Syntax: `associate.objects`

Description: Associates one or more resources.

Availability during this operation: All parent and child objects are available during this operation.

Disassociate Objects

Syntax: `disassociate.objects`

Description: Disassociates one or more resources.

Availability during this operation: All parent and child objects are available during this operation.

Delete Objects

Syntax: `delete.objects`

Description: Deletes a tree of resource objects.

Availability during this operation: The root of the tree for resource objects is available during this operation.

Reassign Primary Parent Object

Syntax: `reassign.primary.parent`

Description: Reassigns primary parent of a resource.

Availability during this operation: The child object and the target parent object is available before and after the operation.

Move Object

Syntax: `move.object`

Description: Moves a resource object to another destination.

Availability during this operation: The source object and the destination resource object to which the source object will be moved is available during this operation.

Move Objects

Syntax: `move.objects`

Description: Moves a tree of resource objects to another destination.

Availability during this operation: The source node of the tree and the destination resource object to which the tree will be moved is available during this operation.

Rename Object

Syntax: `rename.object`

Description: Renames a resource.

Availability during this operation: The object being renamed is available before and after the operation.

Rename Objects

Syntax: `rename.objects`

Description: Renames a self-contained object, which results in the full path updates to the descendant objects.

Availability during this operation: The object being renamed is available before and after the operation.

Copy Object

Syntax: `copy.object`

Description: Copies a resource object to another destination.

Availability during this operation: The source object and the destination resource object to which the source object will be copied is available during this operation.

Copy Objects

Syntax: `copy.objects`

Description: Copies a tree of resource objects to another destination.

Availability during this operation: The source node of the tree and the destination resource object to which the tree will be copied is available during this operation. In the POST phase, the root object ResourceId of the copied tree is available.

Search Objects

Syntax: `query.objects`

Description: A query using the OpenPages GRC API's Query Service.

Availability during this operation: The query string being called.

Search Objects

Syntax: `search.objects`

Description: A legacy event type, this provides access to the `RepositoryService.findResources` call by eliminating objects from the results displayed in Filtered List and object (Detail or Activity) Views. Optionally, you can add additional filters to the call by manipulating the lower level options of the findResources method. This event type does not support the OpenPages GRC API code.

Availability during this operation: Options for the search and its conditions are available.

Example: To display Issue objects owned by the current user only, in Filtered List and object Views, you could add the condition `"Owner=CurrentUser"` to the `findResource` call.

Note: Implement security rules of this type by using the Record Level Security feature, which supersedes the capabilities of Search Objects based triggers. See the IBM OpenPages with Watson Administrators Guide for more information on defining Security Rules.

Configure Event Handlers to Execute Serially

Prior to OpenPages with Watson v7.1, trigger executions were performed serially, since the ability to update objects in parallel didn't exist. In version 7.1, bulk update functionality was implemented in the Grid View. When multiple objects are updated, the update occurs in parallel, which enhances performance.

In some cases, parallel updates can cause inconsistent trigger behavior. For example, if a trigger updates a parent object when all of its sibling objects are in a "specific" state, the update to the parent may not occur if sibling objects are updated in parallel (as sibling object state changes are occurring).

To alleviate this, a configuration mechanism is available to force trigger executions to be performed serially. To enable this behavior, update the trigger configuration file:

1. Check out `_trigger_config_.xml` from the OpenPages repository (see the Configuring Triggers section).
2. Locate the trigger definition that you would like to run serially.
3. Locate the `<eventHandler>` tag.
4. Add a new attribute to this tag: `exclusiveOperationBy="uniqueKeyString"`
   The value `uniqueKeyString` can be any string that uniquely identifies a key (or queue) for trigger serialization. Typically, each trigger will receive its own key. If two triggers might adversely interact if run in parallel, they can use the same `exclusiveOperationBy` key to make both trigger run serially in the same queue. A suggested key format is to use the name of the trigger class. For example, if a trigger class was called com.mycompany.MyTriggerClass, the key name could be com.mycompany.MyTriggerClassKey.
5. Repeat as needed for other triggers.
6. Save the file.
7. Check it back into the OpenPages repository.

Note: Forcing bulk updates to be performed serially may degrade performance.

Implementing a Rule

Each registered trigger will have a rule defined. Every rule is a Java class that extends the DefaultRule. The rule determines whether or not the event applies to the business logic for that trigger. If it does not apply, the event is passed to the next trigger and if no rules apply, the event continues and is processed by the system. The definition of what rules apply is based on the business logic and implementation of the rule class, but common use case may be a rule that applies a "Type match" rule e.g. this rule applies if the event is for an object of type "LossEvent".

To implement a rule, you must extend the following class:

```
com.ibm.openpages.api.trigger.ext.DefaultRule
```

Default rule returns false for all event types. Override the specific isApplicable() methods for events associated with your trigger rule. Example:

```
public boolean isApplicable(CreateResourceEvent event) {
    IResource resource = event.getResource();
    if(resource.isFolder()) {
          //do not apply to folders in this case
          return false;
    } else {
          //do business logic to evaluate if resource applies or not
          return evaluate((IGRCObject)resource);
    }
}
```

Note: The event argument to this method provides a method to retrieve the API Context, which can be used to create the IServiceFactory to use the API Services.

Implementing an Event Handler

Once a rule returns a positive result for an event, one or more event handlers defined for the same trigger will be able to handle the event. Every event handler is a Java class that extends DefaultEventHandler.

To implement an event handler, extend the following class:

```
com.ibm.openpages.api.trigger.ext.DefaultEventHandler
```

Default event handler returns false for all event types. Override the specific handleEvent() methods for events associated with your trigger rule.

Example:

```
public boolean handleEvent(UpdateResourceEvent event) {
    IGRCObject object = (IGRCObject)event.getResource();
    //example: in addition to other updates, set a field to current date
    setCurrentDate(object);
    return true;
}
```

Depending on the type of event, different operations can be performed in your code and to the objects that are contained by the event.

An important method to consider is `throwException.`
The base class for DefaultEventHandler has a utility method throwException, which allows the trigger to throw a formatted exception object and stop the execution of the current operation. For the trigger exception message to be displayed in the UI front-end, it must be thrown in the PRE position event handler.

```
public void throwException(java.lang.String message,
            java.util.List<java.lang.Object> parameters,
            java.lang.Throwable cause,
            Context apiContext)
              throws GRCTriggerException
```

Out-of-the-Box Rules

Content Type Match Rule

Syntax: `com.ibm.openpages.api.trigger.oob.ContentTypeMatchRule`

Description: This rule checks if the object type of the operating object matches the specific value.

Usage: This rule can be used in the following events:

create.object

create.object.with.relations

update.object

delete.objects

associate.objects

disassociate.objects

reassin.primary.parent

move.object

move.objects

rename.object

rename.objects

copy.object

copy.objects

* This rule can be executed in the PRE or POST position, but the rule must be executed only in the PRE position for delete.objects event.

Attributes:

content.type (required)
Object Type Name. E.g. <attribute name="content.type" value="SOXIssue"/>

check.on (optional)	
Determines the scope of the search. E.g. <attribute name="check.on" value="parent"/>. This applies to associate.objects, disassociate.objects, copy.object or copy.objects events only.	
The possible values are:	
parent	Check on the parent only. This applies to associate.objects or disassociate.objects events only.
child	Check on the child only. This applies to associate.objects or disassociate.objects events only. This is default value.
source	Check on the source only. This applies to copy.object or copy.objects events only. This is default value.
destination	Check on the destination only. This applies to copy.object or copy.objects events only.

Detect Property Change Rule

Syntax: `com.ibm.openpages.api.trigger.oob.DetectPropertyChangeRule`

Description: This rule detects when a specified list of fields of a GRC object have changed their value during a create.object or update.object event.

Usage: This rule can be used in following events:

create.object

create.object.with.relations

update.object

* This rule can be executed in the PRE or POST position.

Attributes:

content.type (required)
Resource Type Name. E.g. <attribute name="content.type" value="SOXIssue"/>

fields (required)
Determines the set of fields to detect changes. This attribute is required for all resource based rules. The value is a comma-delimited list of bundle:field. E.g. <attribute name="fields" value= "OPSS-Iss:Assignee,OPSS-Iss:Due Date"/>

check.for (optional)	
Determines the scope of the search.	
The possible values are:	
all	All fields will be checked for changes. This is default value.
any	Checks whether any one field was changed.

Fields Match Rule

Syntax: `com.ibm.openpages.api.trigger.oob.FieldsMatchRule`

Description: This rule checks if values of the operating object match specific values.

Usage: This rule can be used in following events:

create.object

create.object.with.relations

update.object

delete.objects

associate.objects

disassociate.objects

reassign.primary.parent

move.object

move.objects

rename.object

rename.objects

copy.object

copy.objects

* This rule can be executed in the PRE or POST position, but the rule must be executed only in the PRE position for delete.objects event.

Attributes:

content.type (required)
Object Type Name. E.g. <attribute name="content.type" value="SOXIssue"/>

check.on (optional)	
Determines the scope of the search. E.g. <attribute name="check.on" value="parent"/>. This applies to associate.objects, disassociate.objects, copy.object or copy.objects events only.	
The possible values are:	
parent	Check on the parent only. This applies to associate.objects or disassociate.objects events only.
child	Check on the child only. This applies to associate.objects or disassociate.objects events only. This is default value.
source	Check on the source only. This applies to copy.object or copy.objects events only. This is default value.
destination	Check on the destination only. This applies to copy.object or copy.objects events only.

rule.field.nnn (required)
The name of field to be compared. E.g. <attribute name="rule.field.1" value="OPSS-Iss:Additional Description"/> <attribute name="rule.field.2" value="OPSS-Iss:Issue Type"/>

rule.field.value.nnn (required)
The value of field to be compared. <attribute name="rule.field.value.1" value="Test"/> <attribute name="rule.field.value.2" value="Scoping"/> For date data type, the format of value should be in MM/dd/yyyy, e.g. 12/31/2015.

rule.operator.nnn (required)	
The compared operator. E.g. <attribute name="rule.operator.1" value="="/> <attribute name="rule.operator.2" value="="/	
The possible values are:	
=	equals to
!=	not equal to
>	greater than
>=	greater than or equals to
<	less than
<=	less then or equals to

check.for (optional)	
Determines the scope of the compare.	
The possible values are:	
all	Checks whether all fields are matched. This is default value.
any	Checks whether any one field is matched.

Folder Match Rule

Syntax: com.ibm.openpages.api.trigger.oob.FolderMatchRule

Description: This rule checks if the folder path of the operating object matches the specific value.

Usage: This rule can be used in following events:

create.object
create.object.with.relations
update.object
delete.objects
associate.objects
disassociate.objects
reassign.primary.parent
move.object
move.objects
rename.object
rename.objects
copy.object
copy.objects

* This rule can be executed in the PRE or POST position, but the rule must be executed only in the PRE position for delete.objects event.

Attributes:

content.type (required)
Object Type Name. E.g. <attribute name="content.type" value="SOXIssue"/>

folder.path (required)
Determines the path of the folder in which the resource should reside. E.g. <attribute name="folder.path" value="/_op_sox/Project/Default/ICDocumentation/Loss Events"/>

scope (optional)	
The scope of the search.	
The possible values are:	
recursive	Sub-folders will be checked.
self	Only immediate child resources will be checked. This is default value.

check.on (optional)	
Determines the scope of the search. E.g. <attribute name="check.on" value="parent"/>. This applies to associate.objects, disassociate.objects, copy.object or copy.objects events only.	
The possible values are:	
parent	Check on the parent only. This applies to associate.objects or disassociate.objects events only.
child	Check on the child only. This applies to associate.objects or disassociate.objects events only. This is default value.
source	Check on the source only. This applies to copy.object or copy.objects events only. This is default value.
destination	Check on the destination only. This applies to copy.object or copy.objects events only.

Out-of-the-box Handlers

Date Validation Handler

Syntax: com.ibm.openpages.api.trigger.oob.DateValidationHandler

Description: This GRC trigger event handler validates two date fields on an object, throws a validation error message if end date is not after start date.

Usage: This event handler can be used for following events:
create.object
create.object.with.relations
update.object
This event handler must be executed only in the PRE position.

Attributes:

start.date.field (required)
The field name of a date field for start date.

end.date.field (required)
The field name of a date field for end date.

Send Email Event Handler

Syntax: com.ibm.openpages.api.trigger.oob.`SendEmailEventHandler`

Description: A GRC Trigger event handler implementation for sending email out to users specified in the fields of the Resource object.

Usage: This event handler can be used for following events:
create.object
create.object.with.relations
update.object
This handler can be executed in both PRE and POST execution phase. This handler uses the mail server configured in the OpenPages registry as the SMTP server. The following setting must be configured in the OpenPages registry: /OpenPages/Applications/Common/Email/Mail Server.

Attributes:

on.change.only (optional)	
Specifies whether or not an email should be sent out only when the user field has changed.	
The possible values are:	
true	An email will be sent out if the user field has changed.
false	An email will not be sent out. This is the default value.

notify.old.users (optional)
Specifies whether or not an email should be sent out when the old user has changed.

The possible values are:	
true	An email will be sent out if the old user has changed.
false	An email will not be sent out. This is the default value.

from.address (required)
An email address identify where the email come from.

user.fields (required)
A comma-separated list of field paths that contain the user name.

email.subject.string.key (required)
The application text key for the subject of the email.

email.subject.parameter.fields (required)
A comma-separated list of field paths with values that are the parameters to the subject text. The order of the specified fields determines the order of the parameters.

email.body.string.key (required)
The application text key for the body of the email.

email.body.parameter.fields (required)
A comma-separated list of field paths with values that are the parameters to the body text. The order of the specified fields determines the order of the parameters.

email.old.subject.string.key (optional)
The application text key for the subject of the email sent to the old user.

email.old.subject.parameter.fields (optional)
A comma-separated list of field paths with values that are the parameters to the subject text of the email sent to the old user. The order of the specified fields determines the order of the parameters.

email.old.body.string.key (optional)
The application text key for the body of the email sent to the old user.

email.old.body.parameter.fields (optional)
A comma-separated list of field paths with values that are the parameters to the body text of the email sent to the old user. The order of the specified fields determines the order of the parameters.

ignore.failure (optional)	
Specifies whether the email should be continue sent out when there's an error.	
The possible values are:	
true	An email will be continue sent out when there's an error.
false	An email will not be continue sent out when there's an error. This is the default value.

application.url.path (required)
The server information of the OpenPages Application. Required for formating a link to the object. (i.e. "http://:/")

mail.timeout (optional)
The sending email's timeout in seconds. Default is 30 seconds.

mail.retry.times (optional)
The retry times of sending email. Default is 3.

Set Current Date Handler

Syntax: com.ibm.openpages.api.trigger.oob.SetCurrentDateHandler

Description: This GRC trigger event handler sets the value of a date field to the current date.

Usage: This event handler can be used for following events:
create.object
create.object.with.relations
update.object
This event handler must be executed only in the PRE position.

Attributes:

current.date.field (required)
The field name of a date field.

Set Enum Field Handler

Syntax: com.ibm.openpages.api.trigger.oob.SetEnumFieldHandler

Description: This GRC trigger event handler sets the value of an enumerate field to a specific value defined in the attributes.

Usage: This event handler can be used for following events:
create.object
create.object.with.relations
update.object
This event handler must be executed only in the PRE position.

Attributes:

enum.field (required)
The field name of an enumerate field. E.g. <attribute name="enum.field" value="OPSS-Iss:Issue Type"/>

set.value (required)
The system name for the enum value. E.g. <attribute name="set.value" value="Scoping"/>

Trigger Events

The Event objects represents all the information about the operation that is taking place. This is used by rules to determine applicability and the event handlers to perform the automation of business logic. Each

event type has a corresponding class in the com.ibm.openpages.api.trigger.events package, which extends from com.ibm.openpages.api.trigger.events.AbstractEvent.

Events are classified by event type. The TriggerEventType enum defines the possible events that are supported by the trigger framework. All event classes will provide the TriggerEventType, TriggerPositionType and Context. Additionally, depending on the type of event there will be other information relating to the operation taking place, such as IResource of Id for the objects involved. Note: unless otherwise specified, "resource" refers to instances of IResource, which may be either Folders or GRC Objects.

The only non-resource type events that are supported by the framework are for search operations, also referred to as "read triggers", which represent resource retrieval or search operations. For more information, see SearchEvent and QueryEvent.

For more information on the events, refer to the OpenPages GRC API Javadocs.

Disabling Triggers

In certain situations, based on the requirements of the use case, you might want triggers to be disabled. For example, when loading pre-processed data through ObjectManager, you do not want the trigger to fire and perform the same operations as the ones that were performed during pre-processing. In these situations, you can disable triggers.

To disable triggers:

1. Log in to the OpenPages application user interface as a user with administrative privilege for Settings.
2. Go to Administration > Settings menu
3. Expand the Settings tree to find the setting to disable triggers and set to true (Applications | GRCM | Disable Triggers)

Error Handling

The trigger framework allows the display of custom error messages specific to the requirement of the business rule implementation. You can display an error message similar to the following in the UI:

OP-00072: OP needs to have a value to make this object unique within this Member Service.

Where:

- error code `OP-00072` is fixed in the system and is used for all business logic-based errors thrown by a trigger rule or action.
- “OP needs to have a value to make this object unique within this Member Service” is defined in the system as an application text which has “OP” as a parameter.

Key: `com.openpages.<company-abbreviation>.trigger.missing.unique.property`

Text: {0} needs to have a value to make this object unique within this Member Service.

Where: `<company-abbreviation>` represents the abbreviated name of a client company.

In the trigger event handler, you must add the following piece of code to display this error.

```
throwException(“com.openpages.xxx.trigger.missing.unique.property”,
               params,
               null,
               event.getContext());
```

Where:

- `com.openpages.xxx.trigger.missing.unique.property` is the key of the text you want to display for the ‘xxx’ company .
- `params` contains `“OP”` as an element in the list of parameters

Configuring Lifecycle Triggers

A lifecycle trigger definition works on lifecycle enabled objects.

Note: For more information see the “Lifecycles” sections in the *IBM® OpenPages® with Watson New Features Guide*, *User Guide*, and *Solutions Guide*.

The structure of a lifecycle trigger is:

```
<trigger-definitions>
  <grcTrigger name="Name of the Lifecycle trigger 1"
           type="lifecycle">
    <objecttype value="name-of-objecttype"/>
    <lifecyclegroup value=“fieldgroup-with-lifecyclefields”/>
    ….   Email related attributes …
    … transition element(s) …
    … defaultsettings element ….

   … optional eventhandler(s) ……..
  </grcTrigger>
</trigger-definitions>
```

For a lifecycle trigger there is no need to specify the event to handle and the position. The lifecycle trigger will handle the following event and position combinations:

- event=”create.object” position=”pre”
- event=”create.object” position=”post”
- event=”update.object” position=”pre”
- event=”update.object” position=”post”

For a lifecycle trigger there is no need to specify a rule handler or event handler. There are default rule and event handlers. You can however add custom event handlers as specified earlier in the ‘Configuring Triggers’ section. The custom event handlers will be called before the default event handler.

Note: After deploying a new or updating an existing lifecycle trigger XML file, check the OpenPages startup log file for any errors. Check the startup log for possible errors and resolve them to ensure that the trigger functions properly.

Lifecycle trigger Properties

The lifecycle trigger element defines the trigger and contains transition elements, email attributes and event handlers.

name
This represents the name of the GRC Trigger and must be unique across all triggers configured in the system.

type
Set the value of the type attribute to “lifecycle” to specify that this is a lifecycle trigger.

objecttype
Set the value of this element to the OpenPages object type that this lifecycle trigger will be associated with. E.g. “QuestionnaireAssessment”.

lifecyclegroup	
Set the value of this element to the OpenPages field group that contains required fields and was added to the object type specified in the objecttype element. E.g. value=”OPLC-QAssessment”	
The required fields are:	
LCName	Specify which lifecycle to use from enum list
LCStage	Add all stages to possible lifecycle choice from LCName field.
LCStatus	Enumeration of possible lifecycle status depending on stage and transition
LCTransition	Enumerated transitions for all the possible stages from the LCStage field.

Note: The lifecycle trigger handling uses fields that are in another field group named “OPLC-Std”. This field group is also required to be added to the lifecycle enabled object type.

Lifecycle trigger Email Properties

If the default handler should send e-mails when transitions are processed then some attributes need to be added to the lifecycle trigger definition.

The attributes are: from.address, email.subject.string.key, email.subject.parameter.fields, email.body.string.key and email.body.parameters.fields.

See the ‘Send Email Event Handler’ section in this document for information about the attributes.

Lifecycle trigger transition element and Properties

The structure of a lifecycle transition is:

```
<transition name="transition-enumvalue"
         nextstage="stage-enumvalue">
  <setstatus value=" stage-enumvalue"/>
  <assigneefield objecttype=”ojbject type name”
                 field=“fieldgroupname:field name”/>
  <attribute name=”readonly” value=”true|false”/>
  <attribute name=”reviewmode” value=”true|false”/>
  <attribute name=”sendemail” value=”true|false”/>
</transition>
```

There should be one transition element in the lifecycle trigger for each enum value specified in the LCTransition field.
Note: The transition element is only used during a trigger ‘update’ event for both the ‘pre’ and ‘post’ positions.

name
This represents a transition enum value. This should be one of the enum values of the LCTransition field.

nextstage
This represents a stage enum value. This should be one of the enum values of the LCStage field.

setstatus element
The LCStatus field will be set to the value specified. The value should be one of the enum values specified in the LCStatus field.

assigneefield element	
The attributes specified in this element will determine what value should be set in the LCAssignee field on the object. The LCAssignee field is in the OPLC-Std field group.	
objecttype	If the field specified is not on the original object than an objecttype can be used to specify a parent object type to check for the field.
field	The value of the field specified in this attribute will be used to set the LCAssignee field value. The field should be specified in the format – "FieldGroupName:FieldName".

readonly
The value for this attribute can be either 'true' or 'false'. The LCReadOnly field (OPLC-Std field group) on the object is set to the specified value.

reviewmode
The value for this attribute can be either 'true' or 'false'. The LCInReview field (OPLC-Std field group) on the object is set to the specified value.

sendemail
The value for this attribute can be either 'true' or 'false'. If the value is set to 'true', the default event handler will send email to the user(s) in the LCAssignee field. The email sent will use the email attributes specified in the lifecycle trigger. Note: The email is sent during the 'POST' handling of a 'create' or 'update' event.

Lifecycle trigger defaultsettings element and Properties

The structure of a lifecycle defaultsettings is:

```
<defaultsettings nextstage="stage-enumvalue”>
  <setstatus value=" stage-enumvalue"/>
  <assigneefield objecttype=”object type name”
                 field=“fieldgroupname:field name”/>
  <attribute name=”readonly” value=”true|false”/>
  <attribute name=”reviewmode” value=”true|false”/>
  <attribute name=”sendemail” value=”true|false”/>
  <lifecycle name=”lifecycle-enumvalue” nextstage=”stage-enumvalue”>
     <assigneefield objecttype=”object type name”
                    field=“fieldgroupname:field name”/>

  </lifecycle>
</defaultsettings>
```

lifecycle element	
The attributes specified in this element will determine what value should be set in the LCAssignee field on the object. The LCAssignee field is in the OPLC-Std field group.	
name	This represents a lifecycle enum value. This should be one of the enum values of the LCName field.
nextstage	This represents a stage enum value. This should be one of the enum values of the LCStage field.

If there are multiple lifecycles defined for the object – by the LCName enum field having multiple values – then use the lifecycle element in the defaultsettings element to differentiate some settings.

Add a lifecycle element for each lifecycle that needs an assigneefield value that is different from the main defaultsettings section or that requires a different initial stage value.

On a create event of a lifecycle enabled object the trigger handler checks if a lifecycle element has been specified for the current lifecycle – from the LCName enum value; If there is a lifecycle then the attributes specified in the lifecycle element will be used otherwise the attributes in the defaultsettings value will be used.

Note: The lifecycle element is supported starting in OpenPages® 7.3.0.2. And the 'nextstage' attribute of the lifecycle element is supported starting in OpenPages® 8.0.0.1.

There should be only one defaultsettings element in the lifecycle trigger. The defaultsettings is handled the same way as specified above for the transition element.

Note: The 'nextstage' attribute can be used to specify the initial stage when an object is created. This attribute can be used starting in OpenPages® 8.0.0.1

Note: The defaultsettings element is used only during a trigger 'create' event for both the 'pre' and 'post' positions.

Lifecycle trigger setfield element and Properties

Use the setfield element to set values of a field on the object. The setfield element can be added in any transition or defaultsettings element. One or more setfield elements can be added. If an error occurs during the execution the error will be displayed in the UI.

Note: The setfield element can be used starting in OpenPages® 8.0.0.1.

The structure of the setfield element is:

```
  <setfield name="fieldgroupname:field name" value="valuetoset"
append="true|false"/>
```

name
The value for this attribute specifies a field on the object. The value should be specified in the format – "FieldGroupName:FieldName".

value
This represents the value to set on the field.

append
The value for this attribute can be either 'true' or 'false'. The 'append' attribute is optional – if not specified the default value is 'false'. It can be used only for fields of Multi-enum and string data types. If set to true the value(s) will be added to the existing value of the field. In the case of string fields the value will be appended to the existing value.

The supported field types for use in the setfield element are:

- Simple String
- Enum and Multi-enum
- Date
- Boolean
- Integer
- Float

name attribute

The field specified in this attribute will be set with the information specified by the value attribute. Only the 'Description' system field is supported. No field group name is needed for this system field. The field name is case sensitive and should be specified as shown.

e.g. <setfield name="Description" value="...."/>

value attribute

Specifies the value to be set on the field from the name attribute. The value for this attribute will depend on the field's data type.

- The attribute value can specify another field name on the object.

  e.g. value="#OPSS-Iss:Due Date#"

  Note: The other field must be of the same data type.
  Note: Not supported for Enum and multi value enum fields

- The attribute can be specified as a static value.

  e.g. <setfield name="Description" value="This is a test"/>
  <setfield name="OPSS-Iss:Issue Status" value="Closed"/>

- For a date field the value can have the following specific format

  "TODAY" - to set the date field to the current date
  "TODAY+x" - date field will be set to current date plus x number of days
  "TODAY-x" - date field will be set to current date minus x number of days

Lifecycle trigger conditions element and Properties

Use the conditions element to check condition(s) for the transition to complete. Only one conditions element can be added in any transition or defaultsettings element. However, you can nest conditions elements.

A conditions element can contain <condition> elements or other <conditions ../> elements.
If the evaluation of the conditions element is in error state then the transition processing is stopped and an error will be displayed in the UI.

Note: The conditions element can be used starting in OpenPages® 8.0.0.1.

The structure of the conditions element is:

```
  <conditions operation="AND | OR”>
      <condition field=”fieldname” operation=”…”
value=”valuetocompare”/>
      …..
      <attribute name="condition.message.key"
value="applicationtextkey"/>
      <attribute name="condition.message.params"
value="listoffieldparams"/>

  </conditions>
```

conditions	
The conditions element can be used to group one or more condition elements. The evaluation of this conditions element is determined by the operation specified.	
operation	If the value is set to “AND” then all the condition(s) within this element have to evaluate to true, otherwise this conditions element will be in an error state. If the value is set to “OR” then if one of the condition(s) within this element evaluates to true then this conditions element will not be in an error state.
condition element	Specifies an individual condition to evaluate
condition.message.key	The value for this attribute should contain a valid application text key. The application text associated with this key will be used for the error that is logged and displayed in the UI.
condition.message.params	The value for this attribute should contain a comma separated list of fields that are parameters to the application text. The order of the fields specified should be the same as expected in the application text. The values of the fields specified will replace the parameters in the application text.

If the conditions.message.key and conditions.message.params are not specified, default errors will be displayed.

Note: If the application text does not require parameters, the condition.message.params attribute is not needed.

Note: The condition.message.key and condition.message.params should be specified only in the conditions element that is specified at the transition level. They should not be specified in a nested conditions element.

Lifecycle trigger condition element and Properties

The condition element allows a specified field to be checked for a condition. The condition element has to be part of a conditions group element.

During execution each condition element will evaluate to a true or false to indicate whether the condition has been met or not.

The structure of the condition element is:

```
<condition field=”fieldname” operation=”…” value=”valuetocompare”/>
```

field
The value for this attribute specifies a field on the object. The value should be specified in the format – “FieldGroupName:FieldName”.

operation
The value for this specifies the comparison operation. Valid values are: “=”, “!=”, “<”, “>”, “<=”.

value
This represents the value to compare against the field’s value.

The supported field types for use in the condition element are:

- Simple String
- Enum and Multi-value enum
- Date
- Boolean
- Integer
- Float

field attribute

The value of the field specified in this attribute will be compared with the information specified by the value attribute.

Only the ‘Description’ system field is supported. No field group name is needed for this system field. The field name is case sensitive and should be specified as shown.

e.g. <condition field="Description" ..../>

operation attribute

"=" - field value and comparison value should be the same
"!=" - field value and comparison value should not be the same.
"<**"** - field value should be less than the comparison value.
">" - field value should be greater than the comparison value.
"<**="** - field value should be less than or equal to the comparison value.
">=" - field value should be greater than or equal to the comparison value.

Note: The '<' is not a valid character in an attribute so you need to escape it with '<'. This escaped character sequence should be used for the “<” and “<=” operations.

Note: The following operations cannot be used for String, Boolean and Enum data types: “<”, “<=”, “>”, “>=”

value attribute

Specifies the value to compare against the field specified in the field attribute. The value for this attribute will depend on the field’s data type.

- The attribute value can specify another field name on the object.
  e.g. value="#OPSS-Iss:Due Date#"

  Note: The other field must be of the same data type.
  Note: Not supported for Enum and multi value enum fields

- The attribute can be specified as a static value.
  e.g. <condition field="Description" operation=”!=” value=”Initial description”/>

```
< condition field="OPSS-Iss:Issue Status" operation="=" value="Closed"/>
```

- The value can have the following specific format
  "EMPTY" - to compare for an empty field

- The following special formats are for date fields only:

  "TODAY" - to set the date field to the current date
  "TODAY+x" - date field will be set to current date plus x number of days
  "TODAY-x" - date field will be set to current date minus x number of days

Samples

The OpenPages GRC API includes samples which demonstrate code for trigger rules and event handlers, compiling and building the jar containing the custom trigger class files and sample trigger configuration xml. Samples are located in the grc_api/samples/Triggers installation directory. The provided samples demonstrate common use cases and offer developers a starting point for developing new applications.

The sample trigger code includes:

ContentTypeMatchTrigger
: A Rule implementation for matching the Type Definition of a GRC Object.

FolderMatchTrigger
: A Rule implementation for matching the location path of a GRC Object.

FieldsMatchTrigger
: A Rule implementation that extends FolderMatchTrigger and adds the capability to match on one or more Field values.

DetectPropertyChangeTrigger
: A Rule implementation to detect when a field value has changed.

DateValidationAction
: An Event Handler that validates that two dates are correct, relative to each other.

SetCurrentDateAction
: An Event Handler that sets a specified Date Field value to the current date.

SetEnumFieldAction
: An Event Handler that sets a specified Enum type Field value to a value provided in the configuration attributes.

Legacy Out-of-the-Box Rules

Abstract Resource Trigger

Syntax: `com.openpages.sdk.trigger.object.AbstractResourceTrigger`

Description: An abstract rule implementation for operations that involve Resource objects. Trigger rules that apply to resource-based operations should extend this implementation. If the abstract implementation does not satisfy business requirements, you should extend the `AbstractTrigger` class.

This abstract implementation provides easy access to the resource in context, which is based on the operation being performed. Rules that extend this class must implement the following method:

```
boolean isApplicable(OpenpagesSession session, Resource resource)
```

Where: `resource` is the object in context.

This implementation also puts the applicable resource into the trigger context so that it is easily accessible to subsequent rules and actions. To access the applicable resource, use the following code:

```
List<Resource> resources =
(List<Resource>)getContext().getContextAttribute(ResourceTriggerConstants.CON
TEXT_ATTR_APPLICABLE_RESOURCES)
```

The resources in the list are in the order in which these were originally passed to the operation.

The following attributes are optional for all resource based rules:

exclude.folders (optional)	
Determines whether or not the rule applies to folders. This attribute is optional for all operations.	
The possible values are:	
true	Folders will not be processed.
false	Folders will be processed. This is the default value.

check.on (optional)	
Determines the scope of the search. This attribute is currently required for `copy.objects` operations only.	
The possible values are:	
source	Check on the source only. This applies to copy.object and `copy.objects` operation only.
destination	Check on the destination only. This applies to `copy.objects` operations only.
parent	Check on the parent only. This applies to `associate.objects` and `disassociate.objects` operations only.
child	Check on the child only. This applies to `associate.objects` and `disassociate.objects` operations only.

Folder Match Trigger

Syntax: `com.openpages.sdk.trigger.object.FolderMatchTrigger`

Description: An extension of the `AbstractResourceTrigger` that checks whether or not the applicable object resides in a particular folder. Along with the attributes of the abstract implementation above, the following attributes are applicable for this rule:

folder.path (required)
Determines the path of the folder in which the resource should reside. This attribute is required for all resource-based rules.

scope (required)	
The scope of the search. This attribute is required for all resource based rules.	
The possible values are:	
recursive	Sub-folders will be checked.
self	Only immediate child resources will be checked.

Detect Property Change Trigger

Syntax: `com.openpages.apps.common.trigger.object.DetectPropertyChangeTrigger`

Description: An extension of the `FolderMatchTrigger` that detects whether or not the set of fields specified in the configuration has changed on the object. In addition to the attributes of the implementation above, the following attributes are applicable and also apply to this rule:

fields (required)
Determines the set of fields to detect changes. This attribute is required for all resource based rules.

check.for (required)	
Determines the scope of the search. This attribute is required for all resource based rules.	
The possible values are:	
all	All fields will be checked for changes.
any	Checks whether any one field was changed.

Content Type Match Trigger

Syntax: `com.openpages.sdk.trigger.object.ContentTypeMatchTrigger`

Description: An extension of the `AbstractResourceTrigger` that checks whether or not the applicable object is of a particular type. In addition to the attributes of the abstract implementation above, the following attribute applies to this rule:

content.type (required)
Determines the name of content type (i.e., object type). This attribute is required for all resource-based rules.

Legacy Out-of-the-Box Actions

Abstract Resource Trigger Action

Syntax: `com.openpages.apps.common.trigger.object.AbstractResourceTriggerAction`

Description: An abstract implementation of an action for operations that involve Resource objects. Preferably all trigger actions that apply to resource-based operations should extend this implementation. If the abstract implementation does not satisfy the business requirements, you should extend the `AbstractTriggerAction` class.

This abstract implementation provides easy access to the resource in context, which is based on the operation being performed. Actions that extend this class must implement the following method:

```
void processResources(List<Resource> resources)
```

Where: `resources` is the list of resources that were part of the arguments passed to the original SDK method. The resources in the list are in the order in which these were originally passed to the operation.

Usage: This action cannot be used directly. Any extension/implementation of this action can be executed anywhere, in PRE or POST execution phases.

Abstract Change Control Trigger Action

Syntax:
`com.openpages.apps.common.trigger.object.AbstractChangeControlTriggerAction`

Description: An abstract trigger action to detect whether or not any properties of the resource object have changed. If the properties have changed, it will ask the underlying implementation to process the changes.

Actions that extend this class must implement the following method:

```
void handleChange(Resource resource, List<String> modifiedFields)
```

Usage: This action cannot be used directly. Any extension and/or implementation of this action must be executed only in the PRE execution phase because it provides functionality to modify a particular property before the data is persisted.

The following attributes apply to this action:

ignore.fields (required)
A comma-delimited list of property paths of the fields that will be ignored. This attribute is required for all resource-based actions.

Change Control Flag Trigger Action

Syntax: `com.openpages.apps.common.trigger.object.ChangeControlFlagTriggerAction`

Description: An action that extends the `AbstractChangeControlTriggerAction`. It will set the change control flag on a configured property of the resource when there are changes made to the resource object.

Usage: Same as `AbstractChangeControlTriggerAction`.

Along with the `AbstractChangeControlTriggerAction` attribute, the following attributes apply to this action:

change.control.flag (required)
The property path of a single-valued enumeration to use as a change control flag. This attribute is required for all resource-based actions.

change.control.value (required)
The system name of the enumerated value that will be set for the change control field when the resource has been modified. This attribute is required for all resource-based actions.

Abstract Exchange Rate As Of Date Trigger Action

Syntax:
`com.openpages.apps.common.trigger.object.AbstractExchangeRateAsOfDateTriggerAction`

Description: An abstract implementation that allows the exchange rate of a property to be modified.

Actions that extend this class must implement the following method:

```
Date getAsOfDate(Resource resource)
```

This method returns the desired date for the exchange rate.

Usage: This action cannot be used directly. Any extension or implementation of this action should be executed only in PRE execution phase because it provides functionality to modify a particular property before the data is persisted.

The following attributes apply to this action:

currency.field (required)
The property path of the currency field whose exchange rate will be updated.

missing.exchange.rate.as.of.date.key (required)
The application text key for the error message to display when the exchange rate of a given date is missing.

Abstract Picklist Dependency Trigger Action

Syntax:
`com.openpages.apps.common.trigger.object.AbstractPicklistDependencyTriggerAction`

Description: An abstract implementation that processes a set of pick list dependencies on a resource. It will calculate the dependencies and set the selected value into the dependent property of the resource.

Actions that extend this class must implement the following method:

```
List<EnumValId> processPicklistDependency(PropertyType propertyType,
                                          List<EnumValId> currentValues,
                                          List<EnumValId> validValues)
```

This method processes the values provided and determines the values to set on the dependent property. The `currentValues` are the values of the resource in context. The `validValues` are the list of possible values currently configured in the system.

Usage: This action cannot be used directly. Any extension/implementation of this action should be executed only in PRE execution phase because it provides functionality to modify a particular property before the data is persisted.

The following attributes apply to this action:

picklist.dependent.fields (required)
A comma delimited list of dependent pick list property paths.

Picklist Value Mapping Trigger Action

Syntax:
`com.openpages.apps.common.trigger.object.PicklistValueMappingTriggerAction`

Description: An action that extends the `AbstractPicklistDependencyTriggerAction`. It is the simplest form of the value mapping where all the valid values are returned by the `processPicklistDependency` method.

Usage: Same as `AbstractPicklistDependencyTriggerAction`.

Abstract Resource Based Send Email Action

Syntax:
`com.openpages.apps.common.trigger.object.AbstractResourceBasedSendEmailAction`

Description: An abstract action implementation for Resource-based actions to send out email.

Actions that extend this class must implement the following method:

```
Properties getServerProperties()
```

This method retrieves the SMTP server properties.

Usage: This action cannot be used directly. Any extension and/or implementation of this action must be executed in both PRE and POST execution phase.

The following attributes apply to this action:

continue.on.send.failure (optional)	
Specifies whether or not the transaction should be rolled back and reported to the end-user when an error occurs while trying to send an email.	
The possible values are:	
true	Reports to the user when an error occurs. This is the default value.
false	Does not report errors to the user.

Default Send Email To User In Field Action

Syntax:
`com.openpages.apps.common.trigger.object.DefaultSendEmailToUserInFieldAction`

Description: Default implementation for the `SendEmailToUserInFieldAction` implementation. This action requires the 'from' address to be specified as an attribute and uses the mail server configured in the OpenPages registry as the SMTP server. The following setting must be configured in the OpenPages registry:
`/OpenPages/Applications/Common/Email/Mail Server`.

Usage: Same as `AbstractResourceBasedSendEmailAction`.

In addition to the `AbstractResourceBasedSendEmailAction` attribute, the following attributes apply to this action:

from.address (required)
The address from which the email will be sent.

Send Email To User In Field Action

Syntax: `com.openpages.apps.common.trigger.object.SendEmailToUserInFieldAction`

Description: An abstract action implementation for sending email out to users specified in the fields of the Resource object.

Usage: Same as `AbstractResourceBasedSendEmailAction`.

In addition to the `AbstractResourceBasedSendEmailAction` attribute, the following attributes apply to this action:

on.change.only (optional)	
Specifies whether or not an email should be sent out only when the user field has changed.	
The possible values are:	
true	An email will be sent out if the user field has changed.
false	An email will not be sent out. This is the default value.

notify.old.users (optional)	
Specifies whether or not an email should be sent out when the old user has changed.	
The possible values are:	
true	An email will be sent out if the old user has changed.
false	An email will not be sent out. This is the default value.

user.fields (required)
A comma-separated list of field paths that contain the user name.

email.subject.app.string.key (required)
The application text key for the subject of the email.

email.subject.parameter.fields (required)
A comma-separated list of field paths with values that are the parameters to the subject text. The order of the specified fields determines the order of the parameters.

email.body.app.string.key (required)
The application text key for the body of the email.

email.body.parameter.fields (required)
A comma-separated list of field paths with values that are the parameters to the body text. The order of the specified fields determines the order of the parameters.

email.old.subject.app.string.key (optional)
The application text key for the subject of the email sent to the old user.

email.old.subject.parameter.fields (optional)
A comma-separated list of field paths with values that are the parameters to the subject text of the email sent to the old user. The order of the specified fields determines the order of the parameters.

email.old.body.app.string.key (optional)
The application text key for the body of the email sent to the old user.

email.old.body.parameter.fields (optional)
A comma-separated list of field paths with values that are the parameters to the body text of the email sent to the old user. The order of the specified fields determines the order of the parameters.

Set Field Value From Another Field Action

Syntax:
`com.openpages.apps.common.trigger.object.SetFieldValueFromAnotherFieldAction`

Description: An action that extends the `AbstractResourceTriggerAction`. It takes the value of a source field and sets it on the destination field.

Usage: This action must be executed in the PRE execution phase only because it modifies a particular property before the data is persisted. This action does not support setting fields of **Currency** data type. Also, system fields (including name and description) are not supported as source or destination fields.

In addition to the `AbstractResourceTriggerAction` attribute, the following attributes apply to this action:

source.field (required)
The field path of the source field.

destination.field (required)
The field path of the destination field.

reset.source (required)	
Specifies whether or not the source should be set to null.	
The possible values are:	
true	The source will be set to null.
false	The source will not be set to null.

Set Field Value From Parent Action

Syntax: `com.openpages.apps.common.trigger.object.SetFieldValueFromParentAction`

Description: An action that extends the `AbstractResourceTriggerAction`. It takes the value of a source field from the parent object and sets it on the destination field. The parent object is determined by the primary association path.

Usage: This action must be executed in the PRE execution phase only because it modifies a particular property before the data is persisted. This action does not support setting fields of **Currency** data type. Also, system fields (including name and description) are not supported as source or destination fields.

In addition to the `AbstractResourceTriggerAction` attribute, the following attributes apply to this action:

source.type (required)
The name of the object type.

source.field (required)
The field path of the source field.

destination.field (required)
The field path of the destination field.

Abstract Resource Copy Trigger Action

Syntax:
`com.openpages.apps.common.trigger.object.AbstractResourceCopyTriggerAction`

Description: Provides an abstract implementation of a copy based action.

Actions that extend this class must implement one of the following methods:

`void processResource(Resource srcResource, Resource destFolder)`
This method is used for a single object copy.

`void processResources(SetId setId, Resource srcResource, Resource destFolder, String newDestName, CopyOptions options)`
This method is used for a folder hierarchical object copy.

Usage: The class must be extended for actions that are triggered by copy resource operations. No attributes are required by this abstract implementation.

Best Practices

1. The most important point to remember is not to misuse triggers. Note that triggers execute as part of the core functionality of the OpenPages application. Exercise caution when deciding what will be implemented using triggers.

2. Performance, performance, performance!!! Triggers should be thoroughly tested for performance and data integrity before being deployed.

3. Use triggers before (PRE) the execution of the method when:
   - All actions that will calculate and set a new value on the object must be persisted.
   - All validation requirements that will throw an error in the UI.

   Any requirements that perform business logic and will either display an error in the UI or must be persisted before or as part of the original operation should be executed in the PRE phase of the operation. The advantage of using PRE triggers is that data can be processed before it is persisted.

4. Use triggers after (POST) the execution of the method when all actions that requires the information from the current object to be present in the database after the operation for the business logic to work correctly.

5. Always execute triggers in the PRE phase of the operation when triggers will:
   - Change the value of a property on the resource object during create or update of the object.
   - Perform validation of the data entered by the user.

6. Always execute triggers in the POST phase of the operation when triggers will:
   - Perform additional updates on other related objects.
   - Create, move, copy, or lock other objects based on the current object in context.

7. You should not execute Cognos-based computations that depend on the new values of the object properties. These new values are not available to Cognos at the time of trigger execution. These values have not been committed yet to the database.

8. The following utility classes provide utility methods for performing resource based operations. Please refer to the javadoc for more information on these classes and methods.

   Package: com.openpages.apps.service.util
   ObjectNameUtil - For object name related helper methods that provide auto-naming support.
   ObjectProfileUtil - For object provide helper methods.
   ObjectProfileViewUtil - For object profile view helpers.
   ObjectPropertyUtil - For field metadata and object field helpers.
   ObjectSearchUtil - For search helpers.
   ObjectTypeConstants - For constants related to object types.
   ObjectTypeUtil - For object type related helpers.

9. Refer to the OPX SDK and middle-tier services SDK Javadoc for more information on the methods you are planning to use.

Tips

1. Getting a list of modified fields, including system fields, from a Resource object.

```
List<String> modifiedFields = ObjectPropertyUtil.getModifiedFields(resource);
```

2. Getting properties from a Resource object for a given comma-separated property path, in an action.

   Note: a property path = FieldGroupName.PropertyName

```
Property property = super.getPropertyByPathAttribute(triggerAttributeName,
resource)

List<Property> properties =
super.getPropertiesByPathAttribute(triggerAttributeName, resource)

List<Object> values = super.getPropertyValuesByPathAttribute(triggerAttributeName,
resource)

List<String> values =
super.getPropertyValueAsStringsByPathAttribute(triggerAttributeName, resource)
```

3. `ObjectPropertyUtil` provides you with a variety of methods to get the enumeration-related information by name, value or Ids.

4. In the POST phase of a create.object operation, the object that was created is available. This can be used, for example, to retrieve the Resource ID of the new object. To access the newly created object, use the following code:

```
Resource newResource = (Resource)getContext().getReturnedObject()
```

   Alternatively, If you are developing a legacy trigger that extends from `AbstractTriggerAction` you may use the interchangeable `AbstractTriggerAction.getReturnedObject()`.

Legacy Trigger Rules and Actions

Rule Properties

name
This represents the name of the rule and must be unique across all rules configured in the system.

operation	
The operation to which this trigger will be applied.	
The possible values are:	
create.object	Creates an object
create.object.with.relations	Creates an object with parent/child associations
update.object	Updates an object
associate.object	Associates one or more objects
disassociate.object	Disassociates one or more objects
delete.objects	Deletes a tree of objects
reassign.primary.parent	Reassigns primary parent
move.object	Moves an object from one location to another
move.objects	Moves a tree of objects
rename.object	Renames an object
rename.objects	Renames a self-contained object
copy.object	Copies an object from one location to another
copy.objects	Copies a tree of objects

type	
This defines the type of rule.	
The possible values are:	
<oob-trigger-name>	A name of one of the out-of-the-box rules (None, if the rules which are available in the product are available with names).
CUSTOM	For a custom rule implementation

classname
This defines the class name where the business logic of the rule is implemented. This attribute must be specified if the type of rule is CUSTOM. Note that this class has to exist in the runtime class path.

position	
This defines the position where the rule should be executed, whether before or after the operation completes execution.	
The possible values are:	
PRE	Before execution
POST	After execution

Rule Attributes

One or more custom attributes can be defined for rules. These attributes are simple name value pairs and are specific to the rule implementation.

Action Properties

type	
This defines the type of the action.	
The possible values are:	
<oob-trigger-name>	A name of one of the out-of-the-box rules (None if the actions which are available in the product are available with names).
CUSTOM	For a custom action implementation

classname
This defines the class name where the business logic of the action is implemented. This attribute needs to be specified if the type of the action is CUSTOM. Note that this class has to exist in the runtime class path.

Action Attributes

One or more custom attributes can be defined for these actions. These attributes are simple name value pairs and specific to the action implementation.