IBM Security Identity Governance and Intelligence
Version 5.2.3.1

Reference Topics

IBM Security Identity Governance and Intelligence
Version 5.2.3.1

Reference Topics

Table of contents

Table list

Chapter 1. Application programming interfaces

Application programming interfaces (APIs) are part of a plug-in model that you can use to add applications without disrupting existing applications.

Remote application programs run outside of the Identity Governance and Intelligence Java™ virtual machine (JVM). Classes outside of the application packages are not intended to be started by a remote application. Classes in remote applications are documented under the Identity Governance and Intelligence application packages. Server extensions, which run in the Identity Governance and Intelligence JVM, can use any of the classes that are listed in the published API documentation (Javadoc). They are Java classes that run in the same JVM of the caller. These APIs are used to develop Identity Governance and Intelligence customization and extensions that can plug into Identity Governance and Intelligence.

Several application APIs can be started by a remote application. A few server extension APIs in the data services package are also included. The following application APIs are intended to be started by a remote application:

Identity Governance and Intelligence Software Development Kit (SDK)

The Identity Governance and Intelligence SDK can be accessed directly from the Virtual Appliance Dashboard. The SDK contains the following elements.

Table 1. Identity Governance and Intelligence SDK contents

Folders	Contains the following files
`customization`	Files used to customize Identity Governance and Intelligence. For example, adding a custom application in the desk, changing the labels and descriptions of the applications, and setting the date and time format for the entire product". See Customization features.
`javaDocAGCore`	The Javadoc, which provides the documentation for the Identity Governance and Intelligence EJB.
`lib`	The binary versions of the IBM® Security Identity Governance libraries and WebSphere® Application Server client to compile the SDK source.
`Readme`	A `README.txt` file.
`RESTDoc`	Documentation to create REST API calls to the Identity Governance and Intelligence services. See “REST APIs” on page 2.
`RESTExamples`	Examples of the REST API calls.
`src`	The source code of the SDK.
`sas.client.props`	The WebSphere Application Server access configuration information.
`ssl.client.props`	The SSL information.

EJB APIs

The Javadoc includes a set of Java packages that contains:

- The interfaces and methods for managing a certain set of functions, and
- The EJB that these interfaces and methods use

These packages allow a third-party application to establish interoperability with Identity Governance and Intelligence and calls a large set of functions, through EJB technology.

The main package of this set is `com.engiweb.profilemanager.common.interfaces`. It contains:

- The main set of interfaces for the interoperability with the Access Governance Core module
- The Interface ISec API, for managing the authorization function

For more information, see the Identity Governance and Intelligence Javadoc

REST APIs

The REST APIs provide third-party applications some functionality and the interface for operating with Identity Governance and Intelligence. Identity Governance and Intelligence client components send the queries to these REST APIs.

Identity Governance and Intelligence External Authorization Services API
: Accepts or refuses the received request. Use these REST APIs when the RESTful web server returns `3 = WAITING_ASYNCHRONOUS`. The RESTful web service must meet the requirements that are specified in the `IGI External Authorization Services.html` file. Otherwise, external authorization cannot work.

External Authorization Services API
: Manages the request related to a list of permissions or roles that can be added, removed, or renewed according to the request type. See `External Authorization Services.html` for complete information about creating the correct RESTful web service for external authorization.

External SoD APIs
: Checks if the entitlement, group, or user presents Segregation of Duties risks. See `ExtSODServices.html`.

External authorization

Virtual appliance REST APIs
You can develop custom applications with the REST application programming interfaces (APIs) that are supported by IBM Security Identity Governance and Intelligence virtual appliance. The REST APIs are web services that are available so you can administer tasks outside of the virtual appliance user interface.

Identity Brokerage REST APIs
The Identity Brokerage provides a REST API for managing accounts, targets, target profiles, groups membership (modify only), and permissions. The API implements the Simple Cloud Identity Management (SCIM) standard Version 2.0 with custom schema extensions. This implementation enables developers to access and manage identity resources directly by developing client applications that can be invoked from anywhere within the network.

Virtual appliance REST APIs

You can develop custom applications with the REST application programming interfaces (APIs) that are supported by IBM Security Identity Governance and Intelligence virtual appliance. The REST APIs are web services that are available so you can administer tasks outside of the virtual appliance user interface.

The REST APIs are separated into a set of functional components of the virtual appliance. The following list describes the components

Analysis and Diagnostics Monitoring
View information about analysis and diagnostic tools such as SNMP monitoring, storage, CPU usage, memory statistics, event logs, and appliance status.

System Settings Management
View information about export or import settings, network settings, system settings, maintenance, and other aspects.

Configuration Management
View server-setting configuration information, which includes custom files, certificates, mail server, and external entities configuration information about the directory and database servers.

Dashboard
View information about quick links, interfaces, middleware and server monitoring, notifications, partition information, and disk usage.

Download REST API documentation

The REST API documentation for IBM Security Identity Governance and Intelligence virtual appliance is packaged in a compressed file.

Complete these steps.

1. Access http://www.ibm.com/support/docview.wss?uid=swg27046896.
2. Download the `RAPI_DOCS.zip` file to a folder on your local computer.
3. Extract the `RAPI_DOCS.zip` file.
4. Open the `index.html` file to view the REST API documentation.

Identity Governance and Intelligence REST APIs

The Identity Governance and Intelligence platform provides a REST API set for managing the main elements of the data model (users, entitlements, permissions, rights, accounts, and also authorization work-flows and SOD attributes. The API implements the Simple Cloud Identity Management (SCIM) standard (version 2.0), with custom schema extensions. This implementation enables developers to access and manage identity resources directly by developing client applications that can be invoked from anywhere within the network.

Prerequisites

Before to proceed, you must be familiar with the following technologies:

- RESTful API
- JSON (JavaScript Object Notation)
- SCIM specification (RFC7643, RFC7644)

You have also to be familiar with Identity Governance and Intelligence data model.

Restrictions

There is not support for SCIM query filter expressions with `or` operator.

There is not support for using parenthesis or brackets for building query filters.

In a filter string, each attribute must be preceded by the *Universal Resource Name (URN)*.

For example:

urn:ietf:params:scim:schemas:core:2.0:User`:name.givenName co \"James\").`

Currently, the available operators are:

and Boolean operator

eq Operator for comparing if a field of a schema is equal to another entity.

co Operator for checking whether a field of a schema is contained into another entity.

sw Operator for checking whether a string starts with a preset string.

ew Operator for checking whether a string ends with a preset string.

In the current release, some SCIM operations are not supported:

PATCH Not applicable.

BULK Not applicable.

Some SCIM standard attributes are not mapped in IGI data model.

You can find this information looking at the Resource Schema.

In SCIM query, the paging mechanism that is adopted is different from the SCIM specification.

The paging is *page-based*, where the `startPage` field of SCIM Search Request indicates the page target and the `count` field specifies the number of elements in that page.

Supported REST APIs

The following table lists the supported Identity Governance and Intelligence REST APIs.

Table 2. Supported Identity Governance and Intelligence REST APIs

Category	API Name	Resource	Endpoint	Operations	Description
Security API	Login	Security Token	`/igi/v2/security/login`	GET	Log in to Identity Governance and Intelligence system and obtains the token that is needed to call next methods.
	Refresh Token	Security Token	`/igi/v2/security/refresh`	GET	Refresh token. Token expires after N minutes. The value N can be configured through the Virtual Appliance setting.

Table 2. Supported Identity Governance and Intelligence REST APIs (continued)

Category	API Name	Resource	Endpoint	Operations	Description
SCIM API	Resource Schema	Resource Schema	`/igi/v2/schemas/{resourceName}`	GET	Gets resource schema, according to the Universal Resource Name (URN).
	Service Provider Configuration	Service Provider Configuration	`/igi/v2/serviceproviderconfig`	GET	Gets the list of SCIM operations supported, such as filtering, bulk, change password, authentication, and patch. Each operation can be supported or not.
	Resource Discovery	Resource Discovery	`/igi/v2/resourcetype`	GET	Gets list of resources that are provided by server.

Table 2. Supported Identity Governance and Intelligence REST APIs (continued)

Category	API Name	Resource	Endpoint	Operations	Description
Access Governance Core API	Add User	User	/igi/v2/agc/users	POST	Creates the object User that represents the digital identity of a common user of an organization.
	Delete User	User	/igi/v2/agc/users/{userId}	DELETE	Deletes the object User.
	Find User	User	/igi/v2/agc/users/.search	POST	Finds user by a SCIM search request.
	Replace User	User	/igi/v2/agc/users/{userId}	PUT	Updates user information.
	Find User by ID	User	/igi/v2/agc/users/{userId}	GET	Finds a user through the User ID.
	Find User Account	User	/igi/v2/agc/users/{user-id}/accounts/.search	POST	Finds one or more Account objects that are associated to a user through a SCIM search request.
	Change User Account Status	User	/igi/v2/agc/users/accounts/{account_id}/status	PUT	Changes the user account status: disable or enable the account according to specific values.
	Find User Entitlements	Entitlements	/igi/v2/agc/users/{userId}/entitlement/.search	POST	Finds the entitlements of the user through a SCIM search request.
	Find User Rights	Rights	/igi/v2/agc/users/{userId}/entitlement/rights	GET	Finds the list of rights that are associated to an entitlement.
	Add User Entitlement	Entitlement	/igi/v2/agc/users/{userId}/entitlement	POST	Adds an entitlement to a user.
	Remove User Entitlement	Entitlement	/igi/v2/agc/users/{userId}/entitlement/{entId}	DELETE	Removes an entitlement that is associated to a user.
	Add Entitlement	Entitlement	/igi/v2/agc/entitlements	POST	Adds an object Entitlement to the data model.
	Find Entitlement	Entitlement	/igi/v2/agc/entitlements/.search	POST	Finds entitlements through a SCIM search request.
	Delete Entitlement	Entitlement	/igi/v2/agc/entitlements/{entId}	DELETE	Deletes one or more objects of type Entitlement.
	Replace Entitlement	Entitlement	/igi/v2/agc/entitlements/{entitlementId}	PUT	Updates an entitlement.
	Find Account Systems	Account	/igi/v2/agc/accountcfg/.search	POST	Finds all accounts that are registered in Identity Governance and Intelligence.
	Get Password Policy	Password	/igi/v2/agc/accountcfg/password/policy	GET	Gets the password policy that is shared by a set of accounts.
	Change Password	Password	/igi/v2/agc/users/accounts/{account_id}/password	POST	Changes the password of a specific account.
	Check Password	Password	/igi/v2/agc/users/accounts/{account_id}/password/check	POST	Checks if the password specified is compliant with the password policies of a specific account (see Get Password Policy).
	Add Group	Group	igi/v2/agc/hierarchies/{hierarchy_id}/groups/{groupParentId}	POST	Adds an object Group to the data model.
	Delete Group	Group	igi/v2/agc/hierarchies/{hierarchy_id}/groups/{groupId}	DELETE	Deletes an object Group.
	Find Group	Group	/igi/v2/agc/hierarchies/{hierarchy_id}/groups/.search	POST	Specifies the group hierarchy to find. If you do not specify a value, the default value is set to 1, indicating the default group ORGANIZATIONAL_UNIT.
	Find Group Hierarchy	Group	igi/v2/agc/hierarchies/.search	POST	Finds the group hierarchy.
	Replace Group	Group	/igi/v2/agc/hierarchies/{hierarchy_id}/groups/{groupId}	PUT	Update the details of a group.

Table 2. Supported Identity Governance and Intelligence REST APIs (continued)

Category	API Name	Resource	Endpoint	Operations	Description
Access Requests API	Find Workflow	Workflow	/igi/v2/arm/workflows/.search	POST	Finds the workflow IDs available for the logged user.
	Find User by Workflow	User	/igi/v2/arm/{workflowId}/users/.search	POST	Finds users according to a variable set of parameters.
	Find User Entitlement by Workflow	Entitlement	/igi/v2/arm/{workflowId}/users/entitlement/.search	POST	Finds the entitlements that are assigned to a user according to the workflow associated to a user.
	Find Role to Add	Role	/igi/v2/arm/{workflowId}/users/{user_id}/entitlements/.search	POST	According to a selected user, finds roles to be added during the generation of an ARM request.
	Generate User Role Request	Request	/igi/v2/arm/{workflowId}/requests/user/entitlements	POST	Generates request for user role assignment (add roles, remove roles, update roles).
	Find Request to Work	Request	/igi/v2/arm/{workflowId}/requests/.search	POST	Finds the IDs of the requests that are related to the next activities to be processed.
	Find Request Detail	Request	/igi/v2/arm/requests/{request_id}	GET	Gets the details of a request.
	Auth or Exe Request	Request	/igi/v2/arm/requests/{request_id}	POST	If the request is of type AUTH, approve or reject a generated request. If the request is of type EXE, after the approving of the request, an operator makes the needed updates on the target system. Thus, flag the request as "completed".
Separation of Duties API	Check User	Risk	/igi/v2/arc/risks/users/{user_id}	POST	Checks the risks that are associated to a user.
	Check User Full	Risk	/igi/v2/arc/risks/users/{user_id}/full	POST	Checks the full SoD risks that are associated to a user.

Procedure for enabling HTTPS communication

Only HTTPS communication is supported.

See Managing certificates to enable secure communication.

API documentation

To access the REST APIs documentation:

1. Download the `IGI_SCIM_API.zip` file from http://www.ibm.com/support/docview.wss?uid=swg22008422 into a folder on your local computer.
2. Extract the `IGI_SCIM_API.zip` file.
3. Open the `IGI_SCIM_API\Output\index.html` file.

Identity Brokerage REST APIs

The Identity Brokerage provides a REST API for managing accounts, targets, target profiles, groups membership (modify only), and permissions. The API implements the Simple Cloud Identity Management (SCIM) standard Version 2.0 with custom schema extensions. This implementation enables developers to access and manage identity resources directly by developing client applications that can be invoked from anywhere within the network.

Supported REST APIs

The following table lists the Identity Governance and Intelligence supported Identity Brokerage REST APIs.

Note:

- For resource search, limited filtering capability is supported for User and Group resources. All other Identity Brokerage resources do not support filtering.
- For resource search, sorting and pagination are not supported. A search limit is specified in the Identity Brokerage properties file to specify the maximum number of returns that are supported by Identity Brokerage. This search limit applies to all Identity Brokerage managed resources.
- For resource search, the attributes query parameter is supported for User and Group resources to adjust the information that is returned.

Table 3. Supported Identity Brokerage REST APIs

Category	Resource	Endpoint	Operations	Description
Target Profile Management	TargetProfile	`/TargetProfiles`	POST	Loads or updates a target profile that contains metadata for supported targets, including service provider configuration, resource types, and schemas.
	TargetProfile	`/TargetProfileJar`	POST	Loads or updates a connector profile that contains metadata for supported targets, including service provider configuration, resource types, and schemas.
	LanguagePack	`/LanguagePack`	POST	Loads or updates the language pack JAR file that contains labels that are used for localized messages.
	TargetProfile	`/TargetProfiles`	GET	Returns a list of all target profiles that are loaded.
	TargetProfile	`/TargetProfiles/{`*`profile`*`}`	GET	Returns the information for the specified profile.
	Schema	`/TargetProfiles/{`*`profile`*`}/Schema`	GET	Returns the schema definitions for the targets of the specified profile.
Target Management	Target	`/Targets`	POST	Defines a target to Identity Brokerage.

Table 3. Supported Identity Brokerage REST APIs (continued)

Category	Resource	Endpoint	Operations	Description
	Target	/Targets	GET	Returns a list of all targets that are managed by Identity Brokerage.
	Target	/Targets/{*targetId*}	GET	Returns the information, which includes connection status for the specified target.
	ServiceProviderConfig	/Targets/{*id*}/ ServiceProviderConfig	GET	Returns the service provider definition for the specified target.
	Schema	/Targets/{*id*}/Schema	GET	Returns the schema definitions for the specified target.
	Target	/Targets/{*id*}	PUT	Modifies the target information in Identity Brokerage.
	Target	/Targets/{*id*}	DELETE	Deletes the specified target from Identity Brokerage.
User Management	User	/Targets/{*targetId*}/Users	POST	Adds a user to the specified target.
	User	/Targets/{*targetId*}/Users	GET	Returns a list of all users for the specified target.
	User	/Targets/{*targetId*}/Users/ {*userId*}	GET	Returns the information for the specified user.
	Users	/Targets/{*targetId*}/Users/ {*userId*}	PATCH	Adds, modifies, or deletes attribute values for the specified user.
	Users	/Targets/{*targetId*}/Users/ {*userId*}	PUT	Replaces attributes of the specified user with the specified values.
	Users	/Targets/{*targetId*}/Users/ {*userId*}	DELETE	Deletes the specified user from the target.
Group Management	Group	/Targets/{*targetId*}/Groups	GET	Returns a list of all groups for the specified target.
	Group	/Targets/{*targetId*}/ Groups/{*groupId*}	GET	Returns the information for the specified group.
	Group	/Targets/{*targetId*}/ Groups/{*groupId*}	PATCH	Adds, modifies, or deletes attribute values for the specified group.

Table 3. Supported Identity Brokerage REST APIs (continued)

Category	Resource	Endpoint	Operations	Description
		/labels?profile={*profilefile*}&key={*key*}	GET	Returns all the localized labels that are currently available for the specified profile and label key.*
		/Forms/{*profile*}/Target	GET	Returns the information that is needed to create a form to configure a connector for the specified profile type. The information includes the attributes for the connector along with metadata for each attribute.*

Note: The /labels and the /Forms APIs do not implement SCIM v2.0. The differences between the APIs that are not SCIM-compliant as compared to the SCIM-compliant APIs are listed:

- The data that is returned by the non-compliant APIs are not SCIM resources and cannot be managed through the Identity Brokerage.
- The APIs are accessed through the /config context, not the /identity context. For example, https://{host}:{port}/BrokerageService/config/Forms/{profile}/Target
- The response messages that are returned by these APIs are in JSON format, but they are not SCIM-compliant. Therefore, the caller must provide an Accept header of "application/json" instead of "application/scim+json".

Procedure

1. The Identity Brokerage REST API is disabled by default. To enable it, see "Enabling or disabling the Identity Brokerage REST API" on page 11.
2. Set up basic authentication to access REST APIs by creating Identity Brokerage users. See "Managing Identity Brokerage users and passwords" on page 11 to create the authentication credentials.
3. Enable HTTPS communication to the Identity Brokerage. Only HTTPS communication is supported. See Managing certificates to enable secure communication.

   Note:

   - The Identity Brokerage profile uses the same certificate store as the Identity Governance and Intelligence.
   - Use port 8443 for the external client to use the Identity Brokerage REST API. This port is blocked by default. To enable it, see "Enabling or disabling the Identity Brokerage REST API" on page 11.
4. View and run the sample client.

API documentation

To access the REST APIs documentation:

1. Download the `Brokerage Provider SCIM APIs.zip` file from http://www.ibm.com/support/docview.wss?uid=swg27048142 into a folder on your local computer.
2. Extract the `Brokerage Provider SCIM APIs.zip` file.
3. Open the `index.html` file.

Limitations

Attribute values are case-sensitive. When you delete an attribute value, make sure to specify the value in its exact case when it was added to the account. Otherwise, the delete request fails.

Enabling or disabling the Identity Brokerage REST API

The Identity Brokerage REST API is disabled by default. Enable it to use the REST API for managing accounts, groups membership, and permissions.

Procedure

1. Access the command line interface console.
2. To enable:
   a. Enter **igi utilities ib_settings ib_api enable**.
   b. Enter **YES** to confirm the request.
3. To disable:
   a. Enter **igi utilities ib_settings ib_api disable**.
   b. Enter **YES** to confirm the request.

Managing Identity Brokerage users and passwords

Use the **IBPasswordSetter** utility to add, update, or deactivate an Identity Brokerage user and its password from the authentication table.

About this task

Add the Identity Brokerage user with password in the authentication table so that the user can create a client application to communicate with the Identity Brokerage.

Note: If the user is already created, you cannot create it again even if the user is deactivated.

Update the Identity Brokerage password whenever applicable.

Deactivate the Identity Brokerage user and account password if the user no longer needs access to the Identity Brokerage REST APIs.

This procedure is intended only for external users who wants to access the Identity Brokerage REST APIs.

Procedure

1. Access the command line interface console.
2. Enter **igi utilities ib_settings users**.

3. To add an Identity Brokerage user:
   a. Enter **create**.
   b. Enter the user name.
   c. Enter the password.
   d. Re-enter the password for confirmation.
4. To change the Identity Brokerage user password:
   a. Enter **change_password**.
   b. Enter the index corresponding to the user.
   c. Enter the new password.
   d. Re-enter the password for confirmation.

   Note:

   Create or update the password based on the password policy.

   The password must be at least 8 characters. It must contain one upper case, one lower case, one numerical, and one special character. The special character cannot be <, >, `, $, |, ;, or &.
5. To deactivate a user:
   a. Enter **deactivate** to display the list of available Identity Brokerage users.
   b. Enter the index corresponding to the user.
   c. Enter **YES** to confirm the request.
6. To reactivate a user:
   a. Enter **reactivate** to display the list of deactivated Identity Brokerage users.
   b. Enter the index corresponding to the user.
   c. Enter the new password.
   d. Re-enter the password for confirmation.
7. To view the list of available Identity Brokerage users, enter **list**.

Chapter 2. Rules overview

You can use *rules* to define event management that is based on an event type. You can also use rules to automate particular policies.

Configuring rules for Access Risk Controls for SAP

Configuring rules for Access Governance Core
Rules are used to manage different types of events or for the automation of particular policies.

LifeEvents_IN_ORGUNIT_ADD

This flow of rules, named **ORGUNIT_ADD**, adds an organizational unit (OU) to the data model. The flow has only one rule: Add OU.

Rule Class = *Live Events*

Queue = *IN*

Note: If you change the sequence of rules in the flow, the behavior of the flow becomes unpredictable. Only expert administrators can change the sequence.

Add OU

In input, are provided the OU to add and the needed boolean parameters.

You can decide to inherit the roles (false) and the OU resources of the parent OU.

```
when
orgUnit : OrgUnitBean( )
orgUnitErcBean : OrgUnitErcBean( )

then
// [ V1.1 - 2014-05-26 ]
OrgUnitAction.add(sql, orgUnit, false, false);
logger.debug("OU created : " + orgUnit);
```

LifeEvents_IN_ORGUNIT_BEFORE

This flow of rules, named **ORGUNIT_BEFORE**, adds an organizational unit (OU) to the data model, with a check related to the previous presence of the OU and of the parent of the OU. The flow is composed by a sequence of four rules.

Rule Class = *Live Events*

Queue = *IN*

Note: If you change the sequence of rules in the flow, the behavior of the flow becomes unpredictable. Only expert administrators can change the sequence.

Transform ADD to MODIFY

In input, are provided the beans that are related to the OU and Event.

```
when
event : EventBean(  )
   orgUnitErcBean : OrgUnitErcBean(  )

then
// [ V1.1 - 2014-05-26 ]
logger.debug("OU input transformation, event:" + event);

  EventInBean eventin = (EventInBean) event;

  final long ADD = 9;
  final long MODIFY = 10;
  final long DISCARDED = 99;
```

The eventin hosts the event to be managed.

```
  long currentOperation = eventin.getOperation();
  String ouCode = (String) orgUnitErcBean.getAttribute("OU");
  String ouTableId = (String) orgUnitErcBean.getAttribute("ID");

  // ADD when the OU already exists:
  // Transform ADD to MODIFY
  // NOTE: it can happen because OUs created from the console haven't the record in the external table.

  if (currentOperation == ADD) {
    // look for the OU
    OrgUnitBean ouBean = UtilAction.findOrgUnitByCode(sql, ouCode);
    if (ouBean != null) {
      eventin.setOperation(MODIFY);
      // save the real operation code
      eventin.setExtAttr10(Long.toString(currentOperation));

      // allign the orgUnit foreign key to orgUnitErc (attr1)
      ouBean.setAttr1(ouTableId);
      OrgUnitAction.modifyOrgUnit(sql, ouBean);
  }
}
```

Transform MODIFY to ADD

In input, are provided the beans that are related to the OU and Event.

```
when
event : EventBean(  )
   orgUnitErcBean : OrgUnitErcBean(  )

then
// [ V1.1 - 2014-05-26 ]
logger.debug("OU input transformation, event:" + event);

  EventInBean eventin = (EventInBean) event;

  final long ADD = 9;
  final long MODIFY = 10;
  final long DISCARDED = 99;
```

The eventin hosts the event to be managed.

```
  long currentOperation = eventin.getOperation();
  String ouCode = (String) orgUnitErcBean.getAttribute("OU");

  // MODIFY when the OU doesn't exist:
  // Transform MODIFY to ADD

   if (currentOperation == MODIFY) {
     if (ouCode == null) {
```

```
      // In this example we assume that OU CODE is calculates by the ADDs rules
      // Then if OU CODE is null the OU can't exist
      eventin.setOperation(ADD);
      // Save the real operation code
      eventin.setExtAttr10(Long.toString(currentOperation));
    } else {
      // Look for the presence of the OU
      OrgUnitBean ouBean = UtilAction.findOrgUnitByCode(sql, ouCode);
      if (ouBean == null) {
        eventin.setOperation(ADD);
        // Save the real operation code
        eventin.setExtAttr10(Long.toString(currentOperation));
      }
    }
  }
```

Create Temporary Parent OU on ADD

In input, are provided the beans that are related to the OU and Event.

```
when
event : EventInBean(  )
orgUnitErcBean : OrgUnitErcBean( )

then
// [ V1.1 - 2014-05-26 ]

  final int ADD_ORG_UNIT = 9;

  if (event.getOperation() == ADD_ORG_UNIT) {

    String parentCode = (String) orgUnitErcBean.getAttribute("PARENT");
    logger.info("parentCode : "+parentCode );
    OrgUnitBean parentBean = UtilAction.findOrgUnitByCode(sql, parentCode);

    if (parentBean == null) {
      logger.info("parentCode : null");
      OrgUnitBean root = new OrgUnitBean();
      root.setId(1L);
      parentBean = UtilAction.createOrgUnit(sql, parentCode, parentCode, "", root, false, false);

      logger.debug("Parent OU created " + parentCode);
    }
  }
```

Save Event

In input, are provided the OU to add and the needed boolean parameters.

You can decide whether to inherit the roles (false) and the OU resources of the parent OU.

```
when
event : EventBean( state == -1 )

then
// [ V1.1 - 2014-05-26 ]

  event.setState(1l);
```

DeferredEvents_IN_ORGUNIT_ADD

This flow of rules, named **ORGUNIT_ADD**, adds an organizational unit (OU) to the data model. The flow has only one rule: Add OU.

Rule Class = *Deferred Events*

Queue = *IN*

Note: The structure and the content of the flow could be the same as the flow LifeEvents_IN_ORGUNIT_ADD. It's always possible to change the content of the flow for addressing specific needs of the customer.

The flow LifeEvents_IN_ORGUNIT_ADD is run in "real time", while the flow DeferredEvents_IN_ORGUNIT_ADD must be associated to a task that is scheduled with Task Planner.

In this case, see Enabling a flow of rules to be deferred.

DeferredEvents_IN_ORGUNIT_BEFORE

This flow of rules, named **ORGUNIT_BEFORE**, adds an organizational unit (OU) to the data model, with a check related to the previous presence of the OU and of the parent of the OU. The flow is composed by a sequence of four rules.

Rule Class = *Deferred Events*

Queue = *IN*

Note: The structure and the content of the flow could be the same as the flow LifeEvents_IN_ORGUNIT_BEFORE. It's always possible to change the content of the flow for addressing specific needs of the customer.

The flow LifeEvents_IN_ORGUNIT_BEFORE is run in "real time", while the flow DeferredEvents_IN_ORGUNIT_BEFORE must be associated to a task that is scheduled with Task Planner.

In this case, see Enabling a flow of rules to be deferred.

Chapter 3. OUT Queue Events filters

Table 4. OUT events queue filters

Filter	Description
Status	Event status can be one of the following values: • Unprocessed. • Success. • Error.
Operation	Indicates the type of available operation: • Add permissions. • Remove permissions. • Add Delegation • Remove Delegation • Disable User. • Enable User. • Create Account. • Remove Account. • Modify Account. • Change Password. • Add Right. • Remove Right. • Add Resource. • Remove Resource. • Adding Entitlement to User. • Remove Entitlement to User.
Trace	Brief description of the error cause.
Priority	Priority can assume only two values: • Runtime. • Batch. The runtime events have to be processed by the Rule Engine before batch events.
Marker	Marker of the event. It might coincide with the identifier of the target system.
Operation Code	Indicates the customized code of a specific event. The code identifies the entity that has originated the event. For example, if the event is originated by Access Governance Core console, the format of operation code is: PM_<*timestamp*>_*UserID*. PM is a prefix and *UserID* is the name of the logged user.

Table 4. OUT events queue filters (continued)

Filter	Description
Event Start-End Date	Filters defining the time period for searching events.

Chapter 4. OUT Queue Events attributes

Table 5. OUT events queue attributes.

Field	Description
ID	Event identifier.
Account ID	Account identifier that is related to the target system involved in out event.
Master UID	Unique identifier of the user.
Operation	Indicates the type of available operation: • Add permissions. • Remove permissions. • Add Delegation. • Remove Delegation. • Disable User. • Enable User. • Create Account. • Remove Account. • Modify Account. • Change Password. • Add Right. • Remove Right. • Add Resource. • Remove Resource. • Adding Entitlement to User. • Remove Entitlement from User.
Status	Event status can assume only three values: • Unprocessed. • Success. • Error.
ERC Status	The status of the User_ERC table. It can be: • Unprocessed. • Success. • Error.
Trace	Brief description of the error cause.
Detail	Further information on the error that is optionally provided by the external system.
Marker	Marker of the event. It might coincide with the identifier of the target system.
Application	The name of the application that is impacted by the operation/event.
Operation Code	A code that is optionally assigned to the operation.
Free Attribute 1	Sensitive data 1.
Free Attribute 2	Sensitive data 2.
...	...

Table 5. OUT events queue attributes. (continued)

Field	Description
Free Attribute N	Sensitive data N.
Event Date	Indicates the event generation date.
Process Date	Indicates the date in which the event must be processed by the Rule Engine (generally coincides with the event date but can be subsequent if the event processing was postponed).
Priority	Priority can assume only two values: • Runtime. • Batch. The runtime events have to be processed by the Rule Engine before batch events.

Chapter 5. TARGET Queue Account Events filters

Table 6. Target queue - Account events filters.

Filter	Description
Status	Event status can be one of the following values: • Unprocessed. • Success. • Error.
Operation	Indicates the type of available operation: • Add Permission. • Remove Permission. • Reset Password. • Disable user. • Enable user. • Unmatched User. • Modify User. • Create User. • Remove User. • Add Right. • Remove Right. • Entitlement add child. • Entitlement remove child. • Custom.
Operation Code	Indicates the customized code of a specific event. It is selectable only if the **Operation** filter is set on **Custom**.
Account ID	Identifier of the account that is involved in the event.
Marker	Marker of the event. It might coincide with the identifier of the target system.
Trace	Brief description of the error cause.
Process ID	An identifier that is assigned by the target system to one or more events.
Event Start-End Date	Filters defining the time period for searching events.

Chapter 6. TARGET Queue Account Events attributes

Table 7. Target queue - Account events attributes.

Field	Description
ID	The event identifier, a sequential number.
Process ID	An identifier that is assigned by the target system to one or more events. This information can be used when you write rules to identify events or sets of events.
Account ID	The identifier of the account that is involved in the event.
Operation	The operation that is made on the account: • Add Permission. • Remove Permission. • Reset Password. • Disable user. • Enable user. • Unmatched User. • Modify User. • Create User. • Remove User. • Add Right. • Remove Right. • Entitlement adds child. • Entitlement removes child. • Custom.
Status	The event status can be one of these values: • Unprocessed. • Success. • Error.
Trace	A short description of the error.
Detail	Further information on the error that is optionally provided by the external system.
Marker	The marker of the event. It can coincide with the identifier of the target system.
External Reference	Code of the authorization on the target system.
Permission	The permission that was added or removed.
Permission Type	The type of the permission that was added or removed.
Name	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.
Surname	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.
Email	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.
DN	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.

Table 7. Target queue - Account events attributes. (continued)

Field	Description
Display Name	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.
Identity UID	Personal data of the user that is subjected to a `create`, `remove`, `enable`, or `disable user` operation.
Event Date	The event generation date.
Process Date	The date in which the event must be processed by the RE (generally coincides with the event date but can be subsequent if the event processing was postponed).
Ownership	The user who caused the event on the external table.

Chapter 7. TARGET Queue Access Events filters

Table 8. TARGET queue - Access events filters.

Filter	Description
Status	Event status can be one of the following values: • Unprocessed. • Success. • Error.
Operation	Indicates the type of operation that is associated with the event: • Create External Role. • Delete External Role. • Add External Role Child. • Remove External Role Child.
Operation Code	Not used.
Process ID	An identifier that is assigned by the target system to one or more events.
Marker	Marker of the event. It might coincide with the identifier of the target system.
Trace	Brief description of the error cause.
Event Start-End Date	Filters defining the time period for searching events.

Chapter 8. TARGET Queue Access Events attributes

Table 9. Target queue - Access events attributes.

Field	Description
ID	The event identifier, a sequential number.
Process ID	An identifier that is assigned by the target system to one or more events. You can use this information to identify events for writing rules.
Operation	The operation that is associated with the event: • Create External Role. • Delete External Role. • Add External Role Child. • Remove External Role Child.
Status	The event status can be one of these values: • Unprocessed. • Success. • Error.
Trace	A short description of the error.
Marker	The marker of the event. It can coincide with the identifier of the target system.
Master Application	The name of the application that includes the external role that is implicated in the event.
Master name	The name of the created or deleted external role for which the event was started.
Master type	The type of external role. It can be a permission or an external role.
External Ref	
Master entitlement type	The entitlement type identifier of the permission or external role.
Master Description	A short description of the permission or external role.
Child Marker	For events that are involved the addition or removal of an external role child, it is the marker of the event. It can coincide with the identifier of the target system.
Child name	The name of the added or removed external role child for which the event was started.
Child type	The type of the external role child. It can be a permission or an external role.
External Reference	Code of the authorization on the target system.
Child entitlement type	The entitlement type identifier of the external role child.
Child Description	A short description of the external role child.
Event Date	The event generation date.
Process Date	The date in which the event must be processed by the RE (generally coincides with the event date but can be subsequent if the event processing was postponed).
Ownership	The user who caused the event on the external table.

Chapter 9. IN Queue Events filters

Table 10. IN queue - User events filters

Filter	Description
Status	Event status can assume one of the following three values: • Unprocessed. • Success. • Error.
Operation	Indicates the type of operation associated with the event: • Create User. • Modify User. • Remove User. • Move User • Create User (Deferred). • Modify User (Deferred). • Remove User (Deferred). • Move User (Deferred). • Custom.
Operation Code	Not used.
Trace	Brief description of the error cause.
Event Start-End Date	Filters defining the time period for searching events.

Chapter 10. IN Queue Events attributes

Table 11. IN queue - User events attributes.

Field	Description
ID	The event identifier, a sequential number.
User ERC	The identifier of the User_ERC table with the attributes of the user that is processed in the operation/event. The USER_ERC table contains a copy of user data from the external system and matches the PERSON table in the AG Core database.
Operation	The operation that is associated with the event: • Create User. • Modify User. • Remove User. • Move User. • Create User (Deferred). • Modify User (Deferred). • Remove User (Deferred). • Move User (Deferred). • Custom.
Status	The event status can be one of these values: • Unprocessed. • Success. • Error.
Trace	A short description of the error.
Detail	Further information on the error that is optionally provided by the external system.
Identification Number	The ID of the user that is processed in the event.
OU Code	The Organization Unit that is associated with the user that is processed in the event.
Action Type	Field that is used by the external system to add information about the operation.
Action Reason	Field that is used by the external system to add information about the operation.
Event Date	The event generation date.
Process Date	The date in which the event must be processed by the RE (generally coincides with the event date but can be subsequent if the event processing was postponed).
Ownership	The process that owns the event. It is usually the name of the connector that is used with the target system.

Index

R

Printed in USA