IBM® OpenPages® GRC Platform REST API Version 7.0.0

Reference Guide

Copyright Information

Contents

Introduction

The IBM® OpenPages® GRC REST API provides access to OpenPages data and metadata. This data-centric API is specified in terms of resources, their URIs, and actions that can be performed on these URIs. This document describes typical interactions and ways to publish the API in conformance with the Content Management Interoperability Services (CMIS) specification.

Audience

To use the OpenPages GRC REST API guide effectively, you should be familiar with the following:

- OpenPages GRC Platform
- Web services such as ASDL and REST
- HTML and the JavaScript scripting language (JSON)
- Programming languages and integrated development environments (IDEs), such as the Java™ programming language and the Eclipse IDE, or the C# programming language and the Microsoft Visual Studio IDE.

Finding information

To find IBM OpenPages GRC Platform product documentation on the web, including all translated documentation, access the IBM OpenPages GRC Platform Information Center (http://pic.dhe.ibm.com/infocenter/op/v7r0m0/index.jsp). Release Notes are published directly to the Information Center, and include links to the latest technotes and APARs.

RESTful systems

There is abundant literature on REST technology, its core principles, and what makes a system RESTful. Additional reference information is provided at the end of this document. This section provides basic information on RESTful systems.

A resource consists of any application data (or operation data) or metadata that makes up a system and is exposed to the Web as a URI. The URI is the name and address of a resource. If the information doesn't have a URI, it's not a Web resource. The API of a RESTful system is articulated around resources and their URI.

The IBM OpenPages GRC Platform REST API implements a RESTful style server that supports interaction with the client over the HTTP standard protocol. The client-server interaction takes place when the client makes an HTTP request to a URL and provides additional parameters or data in the request's header fields and body. The REST API which will handle and process the request and return an appropriate HTTP response which contains a Status-Code indicating successful operation or failure and optionally some data to be returned to the client in the response body.

Addressability

Addressability is the idea that every object and resource in your system is reachable through a unique identifier. In a RESTful system, addressability is managed through the use of URIs. An application is addressable if it exposes relevant aspects of its dataset as resources. Since resources are exposed through URIs, an addressable application exposes a URI for every piece of information it may serve, including application domain concepts, server operation states, or services.

Statelessness

Another feature of RESTful systems is statelessness, where HTTP requests occur in complete isolation. When the client makes an HTTP request, it includes all the information necessary for the server to fulfill that request. The server never relies on information from previous requests. If that information is important, the client must include it in the request. Statelessness does not mean that the server cannot have state (in fact a server's state should be maintained as resources), rather it means that there is no client state maintained on the server. The client is responsible for maintaining state information relative to its interaction and the server provides state transition information, when required. This principle helps achieve the scalability expected from Web applications.

The URL character set is limited to US-ASCII, which includes a number of reserved characters.

Reserved and restricted characters

Character	Reservation/Restriction
%	Reserved as escape token for encoded characters

Reserved and restricted characters

Character	Reservation/Restriction
/	Reserved for delimiting path segments in the path component
.	Reserved in the path component
..	Reserved in the path component
#	Reserved as the fragment delimiter
?	Reserved as the query-string delimiter
;	Reserved as the parameters delimiter
:	Reserved to delimit the scheme, user/password, and host/port components
$, +	Reserved
@ & =	Reserved; these characters have special meaning in the context of some schemes
{ } | \ ^ ~ [ ] `	Restricted; unsafe handling by various transport agents, such as gateways
< > "	Unsafe; should be encoded because these characters often have meaning outside the scope of the URL, such as delimiting the URL itself in a document.
0x00-0x1F, 0x7F	Restricted; characters within these hexadecimal ranges fall within the nonprintable section of the US-ASCII character set
> 0x7F	Restricted; characters whose hexadecimal values fall within this range do not fall within the 7-bit range of the US-ASCII character set

URL designers have incorporated escape sequences to allow the encoding of arbitrary character values or data using a restricted subset of the US-ASCII character set. The encoding simply represents unsupported characters by an escape notation consisting of a percent sign (%) followed by two hexadecimal characters that represent the ASCII code equivalent. For instance if the requested resource has an identifier that contains one of the reserved characters, it must be URL encoded to escape them.

Addressable space / URI design

In the OpenPages model, the metadata it defines, and their corresponding data can be accessed, updated, and created using the REST API. The REST API will take the form of URIs for these resources. The complete set of supported actions, their paths, parameters and corresponding returned codes are defined. Data sent and returned during these HTTP interactions can either be in the HTTP body or as attachments.

This section will describe the addressable space of an OpenPages application in terms of its URI paths and parameters. The root of an OpenPages application as deployed on an application server should be configurable. By default, /grc/api will be used as the root of every path.

The specification of the URI address and the naming of path segments is not arbitrary. Examples of URI paths for OpenPages REST API

http://localhost/grc/api

http://localhost/grc/api/types

Filter capability

The OpenPages REST API supports retrieving resources using simple filters specified as properties to retrieve, date and time ranges or whether parent-child relationships should be included in the response. Refer to the individual paths described in the Paths and Parameters section for supported filters.

Service Document

A WADL Service document, which describes all the possible paths supported for the OpenPages REST API is provided which supports both HTTP GET and OPTIONS from the root URL http://<SERVER_NAME>:<PORT>/grc/api.
By default, the document is returned as content-type application/vnd.sun.wadl+xml. To retrieve in JSON format, add "Content-Type: application/json" in the request header.

Headers

The OpenPages REST API supports standard HTTP header fields in the request for certain methods.

- For sending data with POST or PUT use `Content-Type: application/json`
- For retrieving with GET use `Accept: application/json`
- `X-HTTP-Method-Override` and `X-Method-Override` header fields are supported to override HTTP PUT and DELETE methods, which may be disallowed by some firewalls.

- `Authorization` header field is supported for all requests when using HTTP Basic authentication. See Security section of this document for more details.

Paths and parameters

The OpenPages GRC REST API supports the following paths and parameters.

API Path Segment	HTTP Method			
	GET	PUT	POST	DELETE
/grc/api/types	X			
/grc/api/types/{typeId}	X			
/grc/api/types/{typeId}/associations	X			
/grc/api/types/{typeId}/associations/parents	X			
/grc/api/types/{typeId}/associations/children	X			
/grc/api/types/{typeId}/associations/{associationId}	X			
/grc/api/contents			X	
/grc/api/contents/{contentId}	X	X		X
/grc/api/contents/{contentId}/document/{fileName}	X	X		
/grc/api/contents/{contentId}/template	X			
/grc/api/contents/{contentId}/action/copy		X		
/grc/api/contents/{contentId}/action/move		X		
/grc/api/contents/{contentId}/action/lock		X		
/grc/api/contents/{contentId}/action/unlock		X		
/grc/api/contents/{contentId}/action/removeAllLocks		X		
/grc/api/contents/{contentId}/associations	X		X	X
/grc/api/contents/{contentId}/associations/parents	X		X	X
/grc/api/contents/{contentId}/associations/parents/{parentId}	X			
/grc/api/contents/{contentId}/associations/children	X		X	X
/grc/api/contents/{contentId}/associations/children/{childId}	X			
/grc/api/contents/{contentId}/auditLogs/association	X			
/grc/api/contents/{contentId}/auditLogs/fields	X			
/grc/api/folders	X		X	
/grc/api/folder/{folderId}	X	X		X
/grc/api/folder/{folderId}/containees	X			

/grc/api/folder/{folderId}/template	X			
/grc/api/workflow/jobs	X		X	
/grc/api/workflow/jobs/{jobId}	X			
/grc/api/workflow/jobs/{jobId}/children	X			
/grc/api/workflow/jobs/{jobId}/owners	X			
/grc/api/workflow/jobs/{jobId}/parent	X			
/grc/api/workflow/jobs/{jobId}/tasks	X			
/grc/api/workflow/jobs/{jobId}/action/{action}		X		
/grc/api/workflow/jobTypes	X			
/grc/api/workflow/jobTypes/{jobTypeId}	X			
/grc/api/workflow/jobTypes/{jobTypeId}/tasks	X			
/grc/api/workflow/jobTypes/{jobTypeId}/owners	X			
/grc/api/workflows/tasks	X		X	
/grc/api/workflows/tasks/{taskId}	X			
/grc/api/workflows/tasks/{tasked}/action/{action}		X		
/grc/api/configuration/adminMode	X	X		
/grc/api/configuration/currrencies	X			
/grc/api/configuration/currencies/base	X			
/grc/api/configuration/currencies/code	X	X		
/grc/api/configuration/currencies/code/exchangeRate	X			
/grc/api/configuration/reportingPeriods	X			
/grc/api/configuration/reportingPeriods/{reportingPeriods}	X			
/grc/api/configuration/reportingPeriods/current	X			
/grc/api/configuration/text/{textKey}	X			
/grc/api/configuration/settings/{settingPath}	X			
/grc/api/configuration/profiles	X			
/grc/api/configuration/profiles/{profile}	X			
/grc/api/configuration/profiles/{profile}/definition	X			
/grc/api/configuration/profiles/fields	X			
/grc/api/security/groups	X		X	
/grc/api/security/groups/{groupId}	X			
/grc/api/security/groups/{groupId}/subgroups	X		X	X
/grc/api/security/groups/{groupId}/users	X		X	X
/grc/api/security/groups/{groupId}/permissions	X			
/grc/api/security/groups/{groupId}/roles	X		X	X

/grc/api/security/users			X	
/grc/api/security/users/{userId}	X	X		
/grc/api/security/users/{userId}/groups	X			
/grc/api/security/users/{userId}/permissions	X			
/grc/api/security/users/{userId}/roles	X		X	X
/grc/api/security/users/{userId}/password		X		
/grc/api/security/permissions	X			
/grc/api/security/permissions/{permissionId}	X			
/grc/api/security/roles	X			
/grc/api/security/roles/{roleId}	X			
/grc/api/security/roles/{rolesId}/access	X			
/grc/api/security/roles/{roleId}/permissions	X			
/grc/api/query/	X		X	

/types

Used to interact with OpenPages metadata. Object Types are the resources provided by /types path and are identified by either name or Id.

- **/grc/api/types**: a GET operation with this path retrieves a list of Object Types for a deployment and their URIs. An individual type could be considered a valid resource accessible through a URI. Use optional parameter includeFieldDefinitions true/false to specify whether or not field definitions should be included in the resource for a type definition. Returns an array of types.

```
[
  {
    "name": "SOXDocument",
    "localizedLabel": "File",
    "description": "OpenPages GRC Object Type",
    "id": "42",
    "rootFolderPath": "/_op_sox_documents/Files and Forms",
    "rootFolderId": "66"
   },
   {
    "name": "SOXExternalDocument",
    "localizedLabel": "Link",
    "description": "OpenPages GRC Object Type",
    "jspPath": "/propertyForm/renderProperties.jsp",
    "id": "46",
    "rootFolderPath": "/_op_sox_documents/Files and Forms",
    "rootFolderId": "66"
```

```
        },
        {
         "name": "SOXIssue",
         "localizedLabel": "Issue",
         "description": "OpenPages GRC Object Type",
         "jspPath": "/propertyForm/renderProperties.jsp",
         "id": "44",
         "rootFolderPath": "/_op_sox/Project/Default/Issue",
         "rootFolderId": "28"
        }
]
```

• **/grc/api/types/{id}:** this URI refers to a specific Object Type uniquely identified by id. A GET with this URI returns a representation of the Object Type including fields.

```
{
    "name": "SOXTask",
    "localizedLabel": "Issue Action Item",
    "description": "OpenPages GRC Object Type",
    "jspPath": "/propertyForm/renderProperties.jsp",
    "id": "7",
    "rootFolderPath": "/_op_sox/Project/Default/IssueActionItems",
    "rootFolderId": "18",
    "fieldDefinitions":
    {
      "fieldDefinition":
      [
        {
          "id": "28",
          "name": "Resource ID",
          "localizedLabel": "Resource ID",
          "dataType": "ID_TYPE",
          "required": true,
          "description": "Resource ID",
          "computed": false,
          "readOnly": true,
          "dependencies":
          {
            "dependency":
            [
            ]
          },
          "enumValues":
          {
            "enumValue":
            [
            ]
          }
        },
        {
          "id": "59",
          "name": "Created By",
          "localizedLabel": "Created By",
          "dataType": "INTEGER_TYPE",
          "required": true,
                "description": "The name of the user who created this object. It is set by the application and
cannot be edited.",
                "computed": false,
                "readOnly": true,
                "dependencies":
                {
                   "dependency":
                   [
                   ]
                },
                "enumValues":
                {
                   "enumValue":
                   [
                   ]
                }
             }
]
```

- **/grc/api/types /{type id}/associations:** These paths will give access to all parent-child associations for the given type as a feed. A GET operation with these paths will return both parent AND child associations for the type identified by id.
- **/grc/api/types /{type id}/associations/[parents|children]:** These paths will give access to parent-child associations defined between available types as a feed. A GET operation with these paths will return the parent OR child associations for the type identified by id.

```
[
   {
      "id": "1",
      "enabled": true,
      "min": 0,
      "max": 2147483647
   },
   {
      "id": "4",
      "enabled": true,
      "min": 1,
      "max": 1
   },
   {
      "id": "6",
      "enabled": true,
      "min": 0,
      "max": 2147483647
   },
   {
      "id": "4",
      "enabled": true,
      "min": 0,
      "max": 2147483647
   },
   {
      "id": "5",
      "enabled": true,
      "min": 0,
      "max": 2147483647
   },
   {
      "id": "7",
      "enabled": true,
      "min": 0,
      "max": 2147483647
   }
]
```

/contents

Used to interact with OpenPages instance data such as GRC Objects. The GRC Objects can be referenced by their Id, or path. The path may be either relative or full path. The path must be URL encoded as it may contain '/' reserved character.

Example: For referencing an instance of an Issue using different options for the "contentId"

Id: 3954

Relative Path: Issue/Test 01/Issue 1

Full Path: /_op_sox/Project/Default/Issue/Test 01/Issue 1.txt

- **/grc/api/contents/{id}**: A GET operation on such a path would return a representation of the OpenPages resource. Given the restrictions on the characters within a URI, the path must be URL-encoded.

  An application data resource with URI template /grc/api/contents/{id} can be modified by performing a PUT operation with the updated value for the resource.

  A DELETE operation would delete the resource.

  Example:

```
{
"name": "Issue 1",
"id": "3954",
"path": "/_op_sox/Project/Default/Issue/Test 01/Issue 1.txt",
"description": "This is an example of free text.",
"parentFolderId": "1969",
"fields":
{
   "field":
   [
      {
         "id": "28",
         "dataType": "ID_TYPE",
         "name": "Resource ID",
         "value": "3954"
      },
      {
         "id": "295",
         "dataType": "ENUM_TYPE",
         "name": "OPSS-Issue:Conclusion",
         "enumValue":
         {
            "id": "380",
            "name": "Deficiency",
                    "localizedLabel": "Deficiency",
                    "index": 1
                 }
              },
              {
                 "id": "288",
                 "dataType": "MULTI_VALUE_ENUM",
                 "name": "OPSS-Issue:Domain",
                 "multiEnumValue":
                 {
                    "enumValue":
                    [
                       {
                          "id": "354",
                          "name": "Operational",
                          "localizedLabel": "Operational",
                          "index": 2
                       },
                       {
                          "id": "355",
                          "name": "Technology",
                          "localizedLabel": "Technology",
                          "index": 3
                       }
                    ]
                 }
              },
              {
                 "id": "284",
                 "dataType": "DATE_TYPE",
                 "name": "OPSS-Issue:Due Date",
                 "value":
                 {
                 }
              },

              {
                 "id": "291",
                 "dataType": "STRING_TYPE",
                 "name": "OPSS-Issue:Identified By Individual",
                 "value": "OpenPagesAdministrator"
              },

           ]
        },
        "typeDefinitionId": "44",
        "primaryParentId": "3049"
     }
```

If the GRCObject referenced by the Id is a SOXDocument type, there will be additional information about the file attachment.

```
 "fileTypeDefinition":
 {
    "fileExtension": "xls",
     "mimeType": "application/vnd.ms-excel",
     "id": "46"
  },
  "contentDefinition":
  {
     "attribute":
     {
        "type": "application/vnd.ms-excel",
        "src": "http://localost/grc/api/contents/131/document/DefaultTemplate.xls"
     }
   }
```

- **/grc/api/contents/{id}/action/copy:** A PUT operation to this URL containing the entry representing the list of GRC objects will copy them to the GRC object specified by the id.

```
{
    id:[{id}],
    options:{
      includeChildren: true,
      conflictBehavior: "CREATECOPYOF",
      typeDefinitions:["SOXBusEntity"]
    }
}
```

  conflictBehavior specifies the desired behavior when a GRCObject with the same name already exists in the target location of a copy.

  There are 4 supported behaviors:-

  CREATECOPYOF: Performs the copy. This operation will prefix the name of the copy in the target location with 'Copy of' for the first instance of a conflict, or 'Copy (n) of' for the nth instance of a conflict.

  OVERWRITE: Overwrites the GRCObject at the destination with the GRCObject from the source.

  USEEXISTING: Keeps the GRCObject at the destination and associates it to the source objects being copied.

  ERROR: Throws OpenPagesException if a conflict exists.

- **/grc/api/contents/{id}/action/move:** A PUT operation to this URL containing the entry representing the list of GRC objects will move them as children of the parent GRC object specified by the id.

```
{
    id:[{id}],
    options:{
      includeChildren : true,
      conflictBehavior: "OVERWRITE",
      typeDefinitions:["SOXBusEntity"]
    }
}
```

conflictBehavior specifies the desired behavior when a GRCObject with the same name already exists in the target location of a move.

There are 3 supported behaviors:

OVERWRITE: Overwrites the GRCObject at the destination with the GRCObject from the source.

USEEXISTING: Keeps the GRCObject at the destination and associates it to the source objects being moved.

ERROR: Throws OpenPagesException if a conflict exists.

Example: A PUT operation to the following URL
http://localhost/grc/api/contents/7917/action/move
containing the entity below will move the GRCObject (with id =7920) as a child of the GRCObject with id =7917.

```
{
    id:[ "7920"],
    options:{
      includeChildren : true,
      conflictBehavior: "OVERWRITE",
      typeDefinitions:["SOXBusEntity"]
    }
}
```

- **/grc/api/contents/{id}/action/lock:** A PUT operation to this URL will lock the GRC Object specified by the id.
  Example: A PUT operation to the following URL will lock the GRC object with id =5220.
  http://localhost /grc/api/contents/5220/action/lock

- **/grc/api/contents/{id}/action/unlock:** A PUT operation to this URL will unlock the GRC Object specified by the id.

Example: A PUT operation to the following URL will unlock the GRC object with id =5220.
http://localhost /grc/api/contents/5220/action/unlock

- **/grc/api/contents/{id}/action/removeAllLock:** A PUT operation to this URL will remove all the locks from the GRC Object specified by the id as well as the hierarchy of children below this GRC Object

- **/grc/api/contents/{id}/document/{document name}:** A GET operation will retrieve the file attachment contents for this document. The content is available by following the "src" attribute URL as well.

Uploading a document

To create (POST) or update (PUT) a new Document object, you must specify the contentDefinition, the attribute which has a type for file mime Type and a children attribute whose value is the contents of the file attachment. For plain text files this would be plain String value representing the contents of the text file being uploaded. For other file types, that are binary. The binary data for the file must be encoded in Base64 and the resulting Base64 string will be the value for the `"children"`:

/grc/api/contents :A POST operation to this URL containing the entry representing the object to be created will create a document.

```
{
    "contentDefinition":
    {
       "attribute":
       {
     "type": "text/plain"           },
       "children": "Updated this document using json put upateDocumentContent API"
    },
    "fileTypeDefinition":
    {
       "id": "9",
       "mimeType": "text/plain",
       "fileExtension": "txt"
    },
    "fields":
    {
      },
  "typeDefinitionId": "4",
  "name": "MytestDoc1.txt",
  "description": "TestDocument created using json"
}
```

- **/grc/api/contents/{ID}/associations:** A GET operation retrieves all associations for the given resources. A POST creates new associations.

```
 [
 {
    "id": "3049",
    "typeDefinitionId": "43",
    "path": "/_op_sox/Project/Default/BusinessEntity/Test 01/Test 01.txt",
    "associationDefinitionId": "5",
    "type": "PARENT"
 },
 {
    "id": "4134",
    "typeDefinitionId": "45",
    "path": "/_op_sox/Project/Default/IssueActionItems/Test 01/Issue Action Item 1.txt",
    "associationDefinitionId": "9",
    "type": "CHILD"
 },
 {
    "id": "32902",
    "typeDefinitionId": "45",
    "path": "/_op_sox/Project/Default/IssueActionItems/Test 01/testActionItem01.txt",
    "associationDefinitionId": "9",
    "type": "CHILD"
 }
]
```

- **/grc/api/contents/{ID}/associations/parents:** A GET operation retrieves only parent associations for the given resources. A POST with a feed for new associations will create new parent associations.
- **/grc/api/contents/{id}/associations?children={id},{id}…;parents={id},{id}…:** A DELETE operation will remove parent and/or child associations for the given resource. The list of parents or children Ids can be of any length. Separate parent and child associations with a semi-colon.
- **/grc/api/contents/{contentId}/auditLogs/association :** A GET operation will return a list of AssociationAuditLogs for the GRCObject specified by the id. If no filter is specified, then all association logs are retrieved. To specify the filters, a query parameter is used.
  /grc/api/contents/{contentId}/auditLogs/association?associationFilter={association Filters}}&startDate={startDate}&endDate={endDate}

Start date and end date is provided in ISO8601 date format.
“yyyMMdd'T'HHmmssZ"

- **/grc/api/contents/{contentId}/auditLogs/fields:** A GET operation to this URL will return a list of FieldAuditLogs for the GRCObject specified by the id. If no filter specified, then all field logs are retrieved. To specify the filters, a query parameter is used /grc/api/contents/{contentId}/auditLogs/fields?fieldFilter={fieldFilters}&startDate={startDate}&endDate={endDate}

Start date and end date is provided in ISO8601 date format yyyyMMdd'T'HHmmssZ
Example: A GET operation to the following URL
http://localhost/grc/api/contents/5220/auditlogs/fields?fieldFilters=Description,Name&startDate=20130101T115227EST&endDate=20130910T115225EST

Returns:

```
[
  {
    "name": "Name",
    "id": "55",
    "newValue": "testObject1updated.txt",
    "oldValue": "testObject1.txt",
    "modifiedBy": "OpenPagesAdministrator",
    "modifiedDate": "2013-09-10T11:52:27.000-04:00"
  },
  {
    "name": "Description",
    "id": "56",
    "newValue": "updated test object  description",
    "oldValue": "test object description",
    "modifiedBy": "OpenPagesAdministrator",
    "modifiedDate": "2013-09-10T11:52:25.000-04:00"
  }
]
```

/grc/api/contents/template?typeId={ID}: A GET operation on /grc/api/contents/template would return a "templated" entry for this type of object specified by the Type Definition Id (typeId) query parameter. It is used as a template to create the new object. Once initialized, the new resource would be POSTed to /grc/api/contents.

```
{
      "fields":
      {
         "field":
         [
            {
               "id": "90",
               "dataType": "DATE_TYPE"
            },
            {
               "id": "300",
               "dataType": "DATE_TYPE"
            },
            {
               "id": "297",
               "dataType": "STRING_TYPE"
            },
            {
               "id": "301",
               "dataType": "STRING_TYPE"
            },
            {
               "id": "299",
               "dataType": "DATE_TYPE"
            },
            {
               "id": "302",
               "dataType": "INTEGER_TYPE"
            },
            {
               "id": "298",
               "dataType": "DATE_TYPE"
            }
         ]
      },
      "typeDefinitionId": "45"
   }
```

- **/grc/api/contents/template?typeId={TypeID}&fileTypeId={FileTypeId}:** For document attachment File Types in addition to the Type Definition Id for the SOXDocument or other type, you must specify a FileTypeId for desired File content type. These file type ids are available in /grc/api/types/{TypeID} for document types.

For TypeId 42 (SOXDocument) and FileTypeId (46, xls file type)

```
   {
      "fields":
      {
         "field":
         [
         ]
      },
      "typeDefinitionId": "42",
      "fileTypeDefinition":
      {
         "fileExtension": "xls",
         "mimeType": "application/vnd.ms-excel",
         "id": "46"
      },
      "contentDefinition":
      {
         "attribute":
         {
            "type": "application/vnd.ms-excel"
         }
      }
   }
```

- **/grc/api/contents/:** A POST to this URL containing the entry representing the object to be created will create the GRC Object. The entry can be constructed from the template returned from /grc/api/contents/template for the type of object being created.

Example: Creating Action Item POST this body to /grc/api/contents with header Content-Type: application/json

```
{
   "fields":
            {
             },
   "typeDefinitionId": "45",
   "primaryParentId": "3954",
    "name": "testActionItem01",           description": "Action Item description"

}
```

/folders

- **/grc/api/folders:** A GET operation returns the root folder "/". The folder information is in the op:folder extension. To view the items contained by this folder, perform a GET operation on the containees. For example, /grc/api/folders/1/containees

```
[
    {
      "name": "_cw_channels",
      "id": "7",
      "path": "/_cw_channels",
      "description": "Channels Folder",
      "parentFolderId": "1"
    },
    {
      "name": "_cw_destination",
      "id": "8",
      "path": "/_cw_destination",
      "description": "Destination Folder",
      "parentFolderId": "1"
    },
    {
      "name": "_op_sox",
      "id": "21",
      "path": "/_op_sox",
      "description": "OpenPages Compliance Objects Folder",
      "parentFolderId": "1"
    },
    {
      "name": "_op_sox_documents",
      "id": "22",
      "path": "/_op_sox_documents",
      "description": "OpenPages Compliance Documents Folder",
      "parentFolderId": "1"
    },
    {
      "name": "Migration Documents",
      "id": "206",
      "path": "/Migration Documents",
      "description": "Contains a history of migration files on this machine",
      "parentFolderId": "1"
    },
    {
      "name": "Reports",
      "id": "30",
      "path": "/Reports",
      "description": "Reporting Component JSPs",
      "parentFolderId": "1"
    },
    {
```

```
    "name": "System",
    "id": "9",
    "path": "/System",
    "description": "System Folder",
    "parentFolderId": "1"
  },
  {
    "name": "Templates",
    "id": "128",
    "path": "/Templates",
    "description": "Application Template Files",
    "parentFolderId": "1"
  },
  {
    "name": "_trigger_config_.xml",
    "id": "41",
    "path": "/_trigger_config_.xml",
    "description": "Trigger Configurations",
    "parentFolderId": "1",
    "fields":
    {
      "field":
      [
        {
          "id": "28",
          "dataType": "ID_TYPE",
          "name": "Resource ID",
          "value": "41"
        },

        {
          "id": "59",
          "dataType": "INTEGER_TYPE",
          "name": "Created By",
          "value": 2
        },
        {
          "id": "58",
          "dataType": "DATE_TYPE",
          "name": "Creation Date",
          "value":
          {
          }
        },
        {
          "id": "60",
          "dataType": "DATE_TYPE",
          "name": "Last Modification Date",
          "value":
          {
          }
        },
        {
          "id": "61",
          "dataType": "INTEGER_TYPE",
          "name": "Last Modified By",
                        "value": 6
                    },
                    {
                        "id": "57",
                        "dataType": "STRING_TYPE",
                        "name": "Location",
                        "value": "/_trigger_config_.xml"
                    },
                    {
                        "id": "55",
                        "dataType": "STRING_TYPE",
                        "name": "Name",
                        "value": "_trigger_config_.xml"
                    }
                ]
            },
            "typeDefinitionId": "1"
        }
]
```

- **/grc/api/folder:** A POST operation **to** this URL containing the entry representing a folder object to be created will create the folder.

```
{
        "name": "testFolder",
        "path": "/testFolder",
        "description": "TestFolder",
        "parentFolderId": "1"
}
```

- **/grc/api/folder/{id}:** A GET operation returns an entry for the specified folder by its id. A PUT operation to this URL containing the entry representing the folder object will update the folder with the specified id. A DELETE operation will delete the folder with the specified id.

Example: GET operation on the following URL

http://localhost/grc/api/folders/7914

Returns:

```
{
"name": "testFolder",
"id": "7914",
"path": "/testFolder",
"description": "TestFolder",
"parentFolderId": "1",
"links":
[
    {
        "href": "7914",
        "rel": "self"
    },
    {
        "type": "application/json",
        "href": "7914?alt=application%2Fjson",
        "rel": "alternate"
    },
        {
          "href": "7914",
          "rel": "edit"
        },
        {
          "type": "application/json",
          "href": "7914?alt=application%2Fjson",
          "rel": "describedby"
        },
        {
          "type": "application/json",
          "href": "1?alt=application%2Fjson",
          "rel": "up"
        },
        {
          "type": "application/json",
          "href": "7914/containees?alt=application%2Fjson",
          "rel": "down"
        }
      ]
    }
```

- **/grc/api/folder/{id}/containees:** A GET operation returns a feed for all children in the folder. Note the entries in some cases will be of folders, but also other resources or GRC Object entries. See /grc/api/contents examples.

/query

- **/grc/api/query?{query specification}**: A GET operation using this URI performs a query using the query service syntax call on the OpenPages model and pages through the result set. The result contains a JSON object with a set of links and an array of rows. Each row contains an object that includes all the fields for each row that was returned from the query. This resource URI supports two query parameters, one for the query string and the other for result pagination support. Query string can be specified using the "q" parameter, the query string syntax that can be used is documented in the *IBM OpenPages GRC Platform API Javadoc*. Pagination is supported using the "skipCount" parameter, which specifies the starting result row. For the first row, "skipCount=0" should be used. For results greater than the default page size of 50, use href URL with attribute rel="next" link to execute the query for the next page.

  Example: A GET operation on the following URL (For unescaped query string: SELECT [Name],[Shared:Shared Integer] FROM [LevelOne])

  http://localhost/grc/api/query?skipCount=0&q=SELECT+%5BName%5D,+%5BShared:Shared+Integer%5D+FROM+%5BLevelOne%5D

  Returns:

```
{
  "links":
  [
     {
        "rel": "self“,
         href":
"http://localhost/grc/api/query?skipCount=0&q=SELECT+%5BName%5D,+%5BShared:Shared+Inte
ger%5D+FROM+%5BLevelOne%5D"
     },
     {
        "rel": "alternate",
 "href":
”http://localhost/grc/api/query?skipCount=0&q=SELECT+%5BName%5D,+%5BShared:Shared+Inte
ger%5D+FROM+%5BLevelOne%5D&alt=application%2Fjson",
        "type": "application/json"
     },
     {
       "rel": "alternate",
       "href":
“http://localhost/grc/api/query?skipCount=0&q=SELECT+%5BName%5D,+%5BShared:Shared+Inte
ger%5D+FROM+%5BLevelOne%5D&alt=application%2Fatom%2Bxml",
        "type": "application/atom+xml"
        },
        {
          "rel": "first",
          "href":
"http://localhost/grc/api/query?skipCount=0&q=SELECT+%5BName%5D,+%5BShared:Shared+Inte
ger%5D+FROM+%5BLevelOne%5D&alt=application%2Fjson",
          "type": "application/json"
        },
        {
          "rel": "next",
          "href":
"http://localhost/grc/api/query?skipCount=50&q=SELECT+%5BName%5D,+%5BShared:Shared+Int
eger%5D+FROM+%5BLevelOne%5D&alt=application%2Fjson",
          "type": "application/json"
        }
      ],
      "rows":
      [
        {
          "fields":
          {
            "field":
            [
              {
                "id": "55",
                "dataType": "STRING_TYPE",
                "name": "Name",
                "value": "Level One 1"
              },
              {
                "id": "142",
                "dataType": "INTEGER_TYPE",
                "name": "Shared:Shared Integer",
                "value": 31
              }
            ]
          }
        },
        {
          "fields":
          {
            "field":
            [
              {
                "id": "55",
                "dataType": "STRING_TYPE",
                "name": "Name",
                "value": "Level One 2"
              },
              {
                "id": "142",
                "dataType": "INTEGER_TYPE",
                "name": "Shared:Shared Integer",
                "value": 17
              }
            ]
        }
      }

    ]
  }
```

- **/grc/api/query** A POST operation using this URI with the query statement and skip count being specified in the body of the request in JSON format performs a query the same as the GET method.

  Example: A POST to the following URL

  http://localhost/grc/api/query

  With JSON Body to define the query:

```
{
  "statement": " SELECT [Resource ID] FROM [SOXDocument] ",
   "skipCount": 0
}
```

  Returns:

```
{
  "links": [
    {
      "rel": "self",
      "href":
"query?skipCount=0&q=SELECT+%5BResource+ID%5D+FROM+%5BSOXDocument%5D"
    },
    {
      "rel": "alternate",
      "href":
"query?skipCount=0&q=SELECT+%5BResource+ID%5D+FROM+%5BSOXDocument%5D&alt=
application%2Fjson",
      "type": "application/json"
    },
    {
      "rel": "first",
      "href":
"query?skipCount=0&q=SELECT+%5BResource+ID%5D+FROM+%5BSOXDocument%5D&alt=
application%2Fjson",
      "type": "application/json"
    }
  ],
  "rows": [
    {
      "fields": {
        "field": [
          {
            "dataType": "ID_TYPE",
            "id": "28",
            "name": "Resource ID",
            "value": "80"
          }
        ]
      }
    }
  ]
}
```

/workflow

/jobs

- **/grc/api/workflow/jobs :** A GET operation returns the core information of all the jobs.
  Example:
  http//localhost/grc/api/workflow/jobs?isOwner=false&state=RUNNING&attachmentIds=1234
- **/grc/api/workflow/jobs:** A POST operation to this URL will submit one or more GRCObjects to a workflow Job type. POST a JSON object containing the Job Type Id or Name, and a List of GRC Object Ids, to submit this Job Type. The Jobs will be queued and started in order.

  Example: Start workflow CJA-Test for Resource with Id 1085
  ```
  {
              "jobtype":"CJA-Test",
              "grcObject":[ "1085" ]
   }
  ```
- **/grc/api/workflow/jobs/{jobId}:** A GET operation returns the core information on the job specified by its id.

- **/grc/api/workflow/jobs/{jobId}/children**: A GET operation returns the list of Children Jobs of the specified job if any exists or “204 No Content code” if no child job exist.
- **/grc/api/workflow/jobs/{jobId}/owners:** A GET operation returns the list of owners of the specified job.
  Example: A GET operation on the following URL
  http://localhost/grc/api/workflow/jobs/2022/owners
  Returns:
  ```
    [
      {
        "userName": "OpenPagesAdministrator",
        "id": "6",
        "description": "System Administrator",
        "firstName": "System",
        "lastName": "Administrator",
  ```

```
      "passwordExpiresInDays": 0,
      "canChangePassword": false,
      "isTemporaryPassword": false,
      "isPasswordChangeFromAdmin": false,
      "isLocked": false,
      "isSecurityAdministrator": false,
      "isHidden": false,
      "isDeleted": false,
      "isEnabled": false,
      "isEditable": false,
      "emailAdress": "OpenPagesAdministrator@openpages.local",
      "adminLevel": 0
    },
    {
      "userName": "OPSystem",
      "id": "2",
      "description": "System User",
      "passwordExpiresInDays": 0,
      "canChangePassword": false,
      "isTemporaryPassword": false,
      "isPasswordChangeFromAdmin": false,
      "isLocked": false,
      "isSecurityAdministrator": false,
      "isHidden": false,
      "isDeleted": false,
      "isEnabled": false,
      "isEditable": false,
      "emailAdress": "OPSystem@openpages.local",
      "adminLevel": 0
    }
  ]
```

- **/grc/api/workflow/jobs/{jobId}/parent:** A GET operation returns the parent job of the specified job.

- **/grc/api/workflow/jobs/{jobId}/tasks:** A GET operation returns the list of tasks associated with the specified job.

  Example: A GET operation on the following URL
  http://localhost/grc/api/workflow/jobs/2022/tasks
  Returns:

```
  [
    {
      "id": "2061",
      "name": "1Submit For Approval",
      "localizedLabel": "1Submit For Approval",
      "description": "1Submit For Approval Description - Assign To: LevelThree.UserSelector",
      "assignees":
      [
        "OpenPagesAdministrator"
      ],
      "state": "ACTIVE",
      "priority": "LOW",
            "jobId": "2022",
            "jobTypeId": "13",
            "choices":
            [
            ],
            "reports":
            [
            ],
            "attachmentIds":
            [
            ],
            "comments":
            [
            ]
        }
    ]
```

- **/grc/api/workflow/jobs/{jobId}/action/{action}:** A PUT operation performs an action on the Job. Terminate is the only action supported.

/jobtypes

- **/grc/api/workflow/jobtypes/{jobTypeId}?typeId={typeId}:** A GET operation returns the job type of the specified id.

  Example: A GET operation on the following URL

  http://localhost/grc/api/workflow/jobtypes?typeId=44

  Returns

```
    [
        {
            "id": "40",
            "name": "Child1",
            "localizedLabel": "Child1",
            "associatedObjectType": "SOXIssue",
            "state": "PUBLISHED"
        }
    ]
```

- **/grc/api/workflow/jobtypes/{jobTypeId}/ owners:** A GET operation returns a List of Users or Groups that are defined as Owners of the specified Job Type

  Example: A GET operation on the following URL

  http://localhost/grc/api/workflow/jobtypes/40/owners

  Returns:

```
    [
        {
            "groupName": "WorkflowJobOwners",
            "id": "9",
            "description": "All Workflow Job Owners",
            "isLocked": false,
            "isHidden": false,
      "isDeleted": false,
      "isEnabled": false,
      "isEditable": false,
      "hasMembers": false,
      "adminLevel": 0
    }
  ]
```

- **/grc/api/workflow/jobtypes/{jobTypeId}/tasks:** A GET operation returns the list of tasks that are defined for the Job Type

  <u>Example:</u> A GET operation on the following URL

  http://localhost/grc/api/workflow/jobtypes/40/tasks

  <u>Returns:</u>

```
  [
    {
      "id": "68",
      "name": "ChildTask1",
      "localizedLabel": "ChildTask1",
      "description": "Task for Child job in Hierarchical Workflow",
      "choices":
      [
      ],
      "reports":
      [
      ]
    }
  ]
```

- **/grc/api/workflow/tasks/{taskId}:** A GET operation returns the core information of the task specified by its id.

  <u>Example:</u> A GET operation on the following URL

  http://localhost/grc/api/workflow/tasks/2061

  <u>Returns:</u>

```
  {
    "id": "2061",
    "name": "1Submit For Approval",
    "localizedLabel": "1Submit For Approval",
    "description": "1Submit For Approval Description - Assign To: LevelThree.UserSelector",
    "assignees":
    [
      "OpenPagesAdministrator"
    ],
    "createDate": "2013-09-19T16:03:08.809-04:00",
    "state": "ACTIVE",
    "priority": "MEDIUM",
    "jobId": "2022",
    "jobTypeId": "13",
    "choices":
    [
      "Request"
    ],
    "reports":
    [
    ],
    "attachmentIds":
    [
       "3714"
    ],
    "comments":
    [
    ]
  }
```

- **/grc/api/workflow/tasks/{taskId}/action/{action}:** A PUT operation performs an action on the Task. Actions supported are: accept, decline, reassign, complete, and comment.

/configuration

/adminMode

- **/grc/api/configuration/adminMode:** A GET operation returns true or false depending on whether the administrator mode is enable or disabled

- **/grc/api/configuration/adminMode?enable{true|false}:** A PUT operation to this URL can modify the system admin mode setting.

/currencies

- **/grc/api/configuration/currencies/base :** A GET operation to this URL will return the OpenPages base currency information.

```
{
   "name": "United States of America, Dollars",
   "id": "10",
   "symbol": "$",
   "precision": 2,
   "isEnabled": true,
   "isoCode": "USD",
   "isBaseCurrency": true
}
```

- **/grc/api/configuration/currencies :** A GET operation to this URL will return all the currency information for all the OpenPages currencies.
- **/grc/api/configuration/currencies/code :** A GET operation to this URL will return the currency information for the specified ISO currency code.

Example: A GET operation to the following URL
http://localhost/grc/api/configuration/currencies/EUR

Returns:

```
{
   "name": "Euro",
   "id": "300",
   "symbol": "€",
   "precision": 2,
   "isEnabled": true,
   "isoCode": "EUR",
   "isBaseCurrency": false
}
```

- **grc/api/configuration/currencies/code?enabled:** A PUT operation using onlyEnabled query parameter to request either all currencies when false or only the enabled currencies when true.

- **grc/api/configuration/currencies/{code}/exchangeRates?date={As of Date}:** A GET operation to this URL will return exchange rates for a currency for a specified ISO currency code. The default path returns the current exchange rate. Optionally, to get the exchange rate as of a particular date, use the date parameter. The expected As of Date format is ISO 8601 'yyyyMMdd'T'HHmmssZ. To retrieve all historical exchange rates for this currency code, omit the date parameter and use parameter 'history=true'.

/reportingPeriods

- **grc/api/configuration/reportingPeriods :** A GET operation to this URL will return all the reporting periods.

```
[
   {
      "id": "1",
      "description": "Finalized reporting period description",
      "creationDate": "2013-08-02T01:14:27.000-04:00",
      "label": "Finalized reporting period name:2013-08-02T01:08:20-04:00",
      "modifiedDate": "2013-08-02T01:14:27.513-04:00",
      "finalizedDate": "2013-08-02T01:14:34.888-04:00"
   },
   {
      "id": "-1",
      "description": "Current Reporting Period",
      "creationDate": "2013-08-01T21:52:26.000-04:00",
      "label": "Current Reporting Period",
      "modifiedDate": "2013-08-01T21:52:26.997-04:00",
      "finalizedDate": "2013-08-01T21:52:26.997-04:00"
   }
]
```

- **grc/api/configuration/reportingPeriods/{reportingPeriod}:** A GET operation to this URL will return the reporting period of the specified name.
- **grc/api/configuration/reportingPeriods/current:** A GET operation to this URL will return the current reporting period

```
{
   "id": "-1",
   "description": "Current Reporting Period",
   "creationDate": "2013-08-01T21:52:26.000-04:00",
   "label": "Current Reporting Period",
   "modifiedDate": "2013-08-01T21:52:26.997-04:00",
   "finalizedDate": "2013-08-01T21:52:26.997-04:00"
}
```

/text

- **grc/api/configuration/text/{textKey}:** A GET operation to this URL will return application text value for the user's locale. The text key is the application text key as defined in the Administration > Application Text menu.

Example: A GET operation to
http://localhost/grc/api/configuration/text/com.label.activityview.detailview
Returns:
Rendered As JSON:

```
[
   "D",
   "e",
   "t",
   "a",
   "i",
   "l",
   " ",
   "V",
   "i",
   "e",
   "w"
]
```

/profiles

- **grc/api/configuration/profiles:** A GET operation to this URL will return the profile summary information of all the profiles.

```
[
  {
    "name": "Default",
    "id": "1",
    "description": "Default Profile",
    "isEnabled": true,
    "isDefault": false,
    "isFallback": false,
    "isDeleted": false
  },
  {
    "name": "OpenPages Platform",
    "id": "2",
    "description": "OpenPages Platform Profile",
    "isEnabled": true,
    "isDefault": false,
    "isFallback": false,
    "isDeleted": false
  },
  {
    "name": "Testing",
    "id": "3",
    "description": "Generated by AFCON 6.1",
    "isEnabled": true,
    "isDefault": true,
    "isFallback": true,
    "isDeleted": false
  }
]
```

- **grc/api/configuration/profiles/{profile}:** A GET operation to this URL will return the profile summary of the specified name or the id.

  Example: A GET operation to the following URL http://localhost/grc/api/configuration/profiles/2

  Returns:

```
{
   "name": "OpenPages Platform",
   "id": "2",
   "description": "OpenPages Platform Profile",
   "isDefault": false,
   "isFallback": false,
   "isEnabled": true,
   "isDeleted": false
}
```

- **grc/api/configuration/profiles/{profile}/definition:** A GET operation to this URL will return full profile definition of the specified name or the id.

  Example: A GET operation to the following URL http://localhost/grc/api/configuration/profiles/3/definitions

  Returns:

```
{
   "name": "OpenPages Platform",
   "id": "2",
   "description": "OpenPages Platform Profile",
   "isDefault": false,
   "isFallback": false,
"objectTypes":
{
   "objectType"
   [
      {
         "name": "SOXBusEntity",
         "id": "43",
         "typeDefinition": "SOXBusEntity",
         "order": 1,
         "fields":
         {
            "field":
            [
               {
                  "fieldSystemName": "Name",
                  "id": "55",
                  "displayType":
                  {
                     "name": "Text",
                     "id": "4",
                     "maxLength": 4000,
                     "columns": 30
                  },
                  "isRequired": true,
                  "isReadOnly": false
               },
               {
                  "fieldSystemName": "Description",
                  "id": "56",
                  "displayType":
                  {
                     "name": "TextArea",
                     "id": "8",
                     "columns": 60,
                     "rows": 5
                  },
                  "isRequired": false,
                  "isReadOnly": false
               }
            ]
         },
         "views":
         {
            "view":
            [

               {
                  "id": "34",
                  "viewTypeName": "Folder",
                  "name": "ID=34 OP=2 AT=43 VT=12",
                  "showDescription": true,
                  "isDefault": false,
                  "isDisabled": false,
                  "attributes":
                  {
                     "attribute":
```

```
        [
        ]
    },
    "rootView": "34",
    "childViews":
    {
        "childView":
        [
        ]
    },
    "fields":
    {
        "field":
        [
            {
                "id": "1",
                "order": 1
            },
            {
                "id": "2",
                "order": 2
            }
        ]
    },
    "sections":
    {
        "section":
        [
        ]
    }
},
{
    "id": "1",
    "viewTypeName": "Overview",
    "showDescription": true,
    "isDefault": false,
    "isDisabled": false,
    "attributes":
    {
        "attribute":
        [
        ]
    },
    "types":
    [
        {
            "type":
            [
                {
                    "id": "43"
                },
                {
                    "id": "46"
                },
                {
                    "id": "42"
```

```
                                    }
                                ]
                            }
                        ]
                    },
                ]
            }
        },
    ]
```

- **grc/api/configuration/profiles/field:** A GET operation to this URL will return the field definitions with profile context for a particular view on an object type e.g.Fields from Detail view for SOXBusEntity
  Example: A GET operation to the following URL http://localhost/grc/api/configuration/profiles/fields/?profile=OpenPages Platform&type=SOXBusEntity&view=Detail

  Returns:

```
[
  {
    "id": "55",
    "name": "Name",
    "localizedLabel": "Name",
    "dataType": "STRING_TYPE",
    "required": true,
    "description": "The name of this object.",
    "computed": false,
    "readOnly": true,
    "dependencies":
    {
      "dependency":
      [
      ]
    },
    "enumValues":
    {
      "enumValue":
      [
      ]
    },
    "isRequired": true,
    "objectOrder": 1,
    "displayType":
    {
      "name": "Text",
      "parameters":
      [
        {
          "key": "spanColumns",
          "value": "false"
        },
        {
          "key": "maxLength",
                "value": "4000"
              },
              {
                "key": "columns",
                "value": "30"
              }
            ]
          },
          "profileFieldDefinitionId": "1",
          "profileTypeDefinitionId": "1"
        },
        {
          "id": "57",
          "name": "Location",
          "localizedLabel": "Folder",
          "dataType": "STRING_TYPE",
          "required": true,
          "description": "The path of this object within this object hierarchy. It cannot be edited after it is
created.",
          "computed": false,
          "readOnly": true,
          "dependencies":
          {
            "dependency":
            [
            ]
          },
          "enumValues":
          {
            "enumValue":
            [
            ]
          },
          "isRequired": true,
          "objectOrder": 3,
          "displayType":
          {
            "name": "Text",
            "parameters":
            [
              {
                "key": "spanColumns",
                "value": "false"
              },
              {
                "key": "maxLength",
                "value": "4000"
              },
              {
                "key": "columns",
                "value": "30"
              }
            ]
          },
          "profileFieldDefinitionId": "3",
          "profileTypeDefinitionId": "1"
```

```
    }
  ]
```

- **grc/api/configuration/settings/{settingPath}**: A GET operation to this URL will return the setting information for the specified setting path. As the path may contain '/' it needs to be an URL encoded

  Example: A GET operation to the following URL

  http://localhost/grc/api/configuration/settings/%2FOpenPages%2FApplications%2FGRCM%2FAudit Trail%2FShow Audit Trail For Custom Forms

  Returns

```
{
   "value": "false",
   "key": "/OpenPages/Applications/GRCM/Audit Trail/Show Audit Trail For Custom Forms"
}
```

/security

/groups

- **/grc/api/security/groups**: **A** POST to this URL containing the entry representing the object to be created will create the user Group. The entry contains **GroupInfo**, which has a groupName, description, emailAddress, parentId, level of the user group and other specifications like whether the group is enabled, editable, hidden or locked.

Example: Create Group

POST this body to /grc/api/security/groups with header Content-Type: application/json

```
{
    "description": "Rest API test group",
     "groupName": "RestTestGroup9",
     "isHidden": false,
     "isEnabled": true,
     "adminLevel": 2,
     "isEditable": false,
     "isLocked": false,
     "isDeleted": false,
     "emailAdress": "others@openpages.local",
        "parentId":"11"

  }
```

- **/grc/api/security/groups:** A GET operation on such a path would return the list of top level groups.
- **/grc/api/security/groups/{id}:** A GET operation on such a path would return a

representation of the OpenPages group.

```
{
   "id": "1019",
   "description": "Rest API test group",
   "groupName": "TestGroup20",
   "isHidden": false,
   "adminLevel": 0,
   "isEnabled": true,
   "isEditable": true,
   "isLocked": false,
   "isDeleted": false,
   "emailAdress": "others@openpages.local",
   "hasMembers": false,
   "links":
   [
      {
         "href": "1019",
         "rel": "self"
      },
      {
         "type": "application/json",
         "href": "1019?alt=application%2Fjson",
         "rel": "alternate"
      },
      {
         "href": "1019",
         "rel": "edit"
      },
      {
         "type": "application/json",
         "href": "1019?alt=application%2Fjson",
         "rel": "describedby"
      },
      {
         "type": "application/json",
         "href": "11?alt=application%2Fjson",
         "rel": "up"
      },
      {
         "type": "application/json",
         "href": "1019/subgroups?alt=application%2Fjson",
         "rel": "down"
      }
   ]
}
```

- An application data resource with URI template /grc/api/groups/{id} can be modified by performing a PUT operation with the updated value for the group.

- **/grc/api/security/groups/{id}/subgroups:** A POST operation to this URL containing the entry representing the list of subgroups or list of subgroups id would add the subgroups to the group with the specified id.

Example: A POST operation on the following URL with header Content-Type: application/json, would make groups with Id 1024 and 1025 members of the group with Id 299 http://localhost/grc/api/security/groups/299/subgroups

```
 [
   {
     id:"1024"

   },
  {
   id:"1025"
  }
]
```

A GET operation to this URL will return the list of subgroups of the group with specified id.

Example: A_GET operation on the following URL
http://localhost/grc/api/security/groups/299/subgroups

Returns:

```
[
  {
    "links":
    [
      {
        "rel": "alternate",
        "href": "../300?alt=application%2Fjson",
        "type": "application/json"
      },
      {
        "rel": "self",
        "href": "../300"
      },
      {
        "rel": "edit",
        "href": "../300"
      },
      {
        "rel": "describedby",
        "href": "../300?alt=application%2Fjson",
        "type": "application/json"
      },
      {
        "rel": "up",
        "href": "../299?alt=application%2Fjson",
        "type": "application/json"
      },
            {
              "rel": "down",
              "href": "../300/subgroups?alt=application%2Fjson",
              "type": "application/json"
            }
        ],
        "groupName": "API_Sub_Group75bc19db-6150-44a7-92e6-ef21d12f26b4",
        "id": "300",
        "description": "Sub group created for CRUD test through API - updated",
        "isLocked": false,
        "isHidden": false,
        "isDeleted": false,
        "isEnabled": false,
        "isEditable": true,
        "hasMembers": false,
        "emailAdress": "foo@ibm.com",
        "adminLevel": 0
    }
]
```

- **/grc/api/security/groups/{id}/subgroups?subgroups={subgroupId}:** A DELETE operation would remove specified subgroups from the group with the specified id.

Example: A DELETE operation to the following URL will remove the subgroups with ids 1024 and 1025 from the group with id 1020.
http://localhost /grc/api/security/groups/1020/subgroups?subgroups=1024,1025

- **/grc/api/security/groups/{id}/users**: A POST to this URL containing the entry representing the user objects will add the users to the Group specified by {Id}. The entry can contain ids of user objects or more information, such as username, firstName, lastName, and localeISOCode.

Example: Adding users to a group
POST this body to /grc/api/security/groups/{Id}/users with header Content-Type: application/json

```
[
     {
              id:"2012"
    },
    {
              id:"2008",
              userName:"RESTUSER2",
              firstName:"user2",
              lastName:"test",
              "localeISOCode": "en_us",
    }
  ]
```

- **/grc/api/security/groups/{Id}/users:** A GET operation on such a path would return the list of all the users that belong to the group.

A DELETE operation would delete the specified users from the group with the specified Id.

Example: A DELETE operation to the following URL will remove the user with id 1021 from the group with id 1020.

http://localhost /grc/api/security/groups/1020/users?users=1021

- **/grc/api/security/groups/{Id}/roles :** A GET operation to this URL will return the list of roles associated with the specified group id.
- **/grc/api/security/groups/{Id}/roles?role={roleId}&context={grcObjectId}:** A POST operation to this URL will assign the roles to the specified group id.

Example:

http://localhost/grc/api/security/groups/2010/roles?role=2033&context=14601

A DELETE operation to this will revoke the specified roles of the group with the specified Id.

/permissions

- **/grc/api/security/permissions:** A GET operation to this URL returns a list of all the application permissions.
- **/grc/api/security/permissions/{permissionId}:** A GET operation to this URL returns the application permission for the specified Id.

Example**:** A GET operation to http://localhost/grc/api/security/permissions/3

Returns:

```
{
   "name": "System Administration Mode",
   "id": "3",
   "description": "System administration mode functions",
   "fullName": "All/Administration/System Administration Mode"
}
```

/roles

- **/grc/api/security/roles:** A GET operation to this URL returns the list of all role templates.

  Returns:

```
[
   {
      "id": "241",
      "description": "used in role template related automation tests for REST API.  ",
      "roleName": "RiskApiTestRoleTemplate_1",
      "isEnabled": true,
      "isLocked": false,
      "grcObjectType": "SOXBusEntity",
      "dateCreated": "2013-09-23T12:19:03.680-04:00",
      "dateChanged": "2013-09-25T22:49:52.285-04:00"
    }
  ]
```

- **/grc/api/security/roles/{roleId}:** A GET operation to this URL returns the role template for the specifed role.
- **/grc/api/security/roles/{roleId}/access:** A GET operation returns the list of all the access permission associated with the specified role.

Example**:** A GET operation to the following URL
http://localhost/grc/api/security/roles/241/access

Returns:

```
  [
    {
      "id": "1",
      "canRead": true,
      "canWrite": false,
      "canDelete": false,
      "canAssociate": false,
      "rootFolderId": "14",
      "objectTypeId": "5"
    },
    {
      "id": "2",
      "canRead": true,
      "canWrite": false,
      "canDelete": false,
      "canAssociate": false,
      "rootFolderId": "17",
      "objectTypeId": "5"
    }
  ]
```

- **/grc/api/security/roles/{roleId}/permissions:** A GET operation to this URL returns the list of all the permissions associated with the specified role.

```
  [
     {
       "name": "All Permissions",
       "id": "1",
      "fullName": "All"
     },
    {
      "name": "Administration",
      "id": "2",
      "description": "Access all administrative functions",
      "fullName": "All/Administration"
     },
    {
        "name": "System Administration Mode",
       "id": "3",
        “description": "System administration mode functions",
       "fullName": "All/Administration/System Administration Mode"
     },
     {
       "name": "Files",
        "id": "4",
     "fullName": "All/Files"
     }
  ]
```

- /users/**grc/api/security/users:** A POST operation to this URL containing the entry representing the object to be created will create the user. The entry contains user information for the user to create, which has username, description, first and last name of the user, email address and other specifications like user locale, group memberships and whether the user is enabled or hidden.

  Example: POST this body to /grc/api/security/users with header Content-Type: application/json

```
{
  userName:"RESTUSER1",
  firstName:"user1",
  lastName:"test",
  localeISOCode: "en_us",
  emailAdress: "others@openpages.local",
  groups:
   {
            group:
            [
                  {
                        id :"2003"
                        groupName:"RestTestGroup1"
                   },
                  {
                        id :"2004" ,
                        groupName:"RestTestGroup2"
                  }
            ]
  }

}
```

- **/grc/api/security/users/{userId}:** A GET operation to this URL will return the user information associated with the specified user Id.

```
{
   "id": "1021",
   "userName": "RUSER1",
   "adminLevel": 0,
   "isEnabled": true,
   "isHidden": false,
   "isEditable": true,
   "firstName": "Ruser1",
   "lastName": "Rtest",
   "passwordExpiresInDays": 0,
   "isTemporaryPassword": false,
   "isPasswordChangeFromAdmin": false,
   "isLocked": false,
   "isDeleted": false,
   "emailAdress": "others@openpages.local",
   "localeISOCode": "en_us",
   "canChangePassword": true,
   "passwordCreationDate": "2013-09-25T22:01:47.408-04:00",
   "isSecurityAdministrator": true
}
```

- **/grc/api/security/users/{userId}:** A PUT operation to this URL containing the entry representing the object to be updated will update the user information associated with the specified user Id.

Example: A PUT with the following to http://localhost/grc/api/security/users/1021

```
{
    "id": "1021",
    "userName": "RUSER1",
    "description":"Rest API test user 1"
}
```

Returns:

```
{
    "id": "1021",
    "description": "Rest API test user 1",
    "userName": "RUSER1",
    "adminLevel": 0,
    "isEnabled": false,
    "isHidden": false,
    "isEditable": true,
    "firstName": "Ruser1",
    "lastName": "Rtest",
    "passwordExpiresInDays": 0,
    "isTemporaryPassword": false,
    "isPasswordChangeFromAdmin": false,
    "isLocked": false,
    "isDeleted": false,
    "emailAdress": "others@openpages.local",
    "localeISOCode": "en_us",
    "canChangePassword": false,
    "passwordCreationDate": "2013-09-25T22:04:44.744-04:00",
    "isSecurityAdministrator": true
}
```

- **/grc/api/security/users/{userId}/groups :** A GET operation to this URL will return the list of groups associated with the specified user Id.

Returns:

```
[
  {
    "id": "1020",
    "description": "Rest API test group",
    "groupName": "Group1",
    "adminLevel": 0,
    "isEnabled": true,
    "isHidden": false,
    "isEditable": true,
    "isLocked": false,
    "isDeleted": false,
    "emailAdress": "others@openpages.local",
    "hasMembers": false
  },
  {
    "id": "3",
    "description": "All OpenPages Users",
    "groupName": "OpenPages",
    "adminLevel": 2,
    "isEnabled": true,
    "isHidden": false,
    "isEditable": false,
    "isLocked": false,
    "isDeleted": false,
    "emailAdress": "AllOpenPagesUsers@openpages.local",
    "hasMembers": false
  }
]
```

- **grc/api/security/users/{userId}/permissions:** A GET operation to this URL will return the list of all the application permission associated with the specified user Id.
- **/grc/api/security/users/{userId}/roles:** A GET operation to this URL will return the list of roles associated with the specified user Id.
- **/grc/api/security/users/{userId}/roles?role={roleId}&context={grcObjectId}:** A POST operation to this URL will assign the roles to the specified user id. A DELETE operation to this will revoke the specified roles of the user id.
- **/grc/api/security/users/{userId}/password:** A PUT operation to this URL containing the entry representing the object to be updated will reset the password of the specified user id.

Appendix: Return codes

This section documents high-level guidelines for interpreting HTTP return status codes:

1. 200 (“OK”) should be used to indicate nonspecific success.

   In most cases, 200 is the code the client hopes to see. It indicates that the REST API successfully carried out whatever action the client requested, and that no more specific code in the 2xx series is appropriate. Unlike the 204 status code, a 200 response should include a response body.

2. 200 (“OK”) must not be used to communicate errors in the response body.

   Always make proper use of the HTTP response status codes as specified by the rules in this section. In particular, a REST API must not be compromised in an effort to accommodate less sophisticated HTTP clients.

3. 201 (“Created”) must be used to indicate successful resource creation.

   A REST API responds with the 201 status code whenever a collection creates, or a store adds, a new resource at the client’s request. There may also be times when a new resource is created as a result of some controller action, in which case 201 would also be an appropriate response.

4. 202 (“Accepted”) must be used to indicate a successful start of an asynchronous action.

   A 202 response indicates that the client’s request will be handled asynchronously. This response status code tells the client that the request appears valid, but it still may have problems once it’s finally processed. A 202 response is typically used for actions that take a long while to process.

   Controller resources may send 202 responses, but other resource types should not.

5. 204 (“No Content”) should be used when the response body is intentionally empty.

   The 204 status code is usually sent out in response to a PUT, POST, or DELETE request, when the REST API declines to send back any status message or representation in the response message's body. An API may also send 204 in conjunction with a GET request to indicate that the requested resource exists, but has no state representation to include in the body.

6. 301 ("Moved Permanently") should be used to relocate resources.

   The 301 status code indicates that the REST API's resource model has been significantly redesigned and a new *permanent* URI has been assigned to the client's requested resource. The REST API should specify the new URI in the response's Location header.

7. 302 ("Found") should not be used.

   The intended semantics of the 302 response code have been misunderstood by programmers and incorrectly implemented in programs since version 1.0 of the HTTP protocol. The confusion centers on whether it is appropriate for a client to always automatically issue a follow-up GET request to the URI in response's Location header, regardless of the original request's method. For the record, the intent of 302 is that this automatic redirect behavior only applies if the client's original request used either the GET or HEAD method.

   To clear things up, HTTP 1.1 introduced status codes 303 ("See Other") and 307 ("Temporary Redirect"), either of which should be used instead of 302.

8. 303 ("See Other") should be used to refer the client to a different URI.

   A 303 response indicates that a controller resource has finished its work, but instead of sending a potentially unwanted response body, it sends the client the URI of a response resource. This can be the URI of a temporary status message, or the URI to some already existing, more permanent, resource.

   Generally speaking, the 303 status code allows a REST API to send a reference to a resource without forcing the client to download its state. Instead, the client may send a GET request to the value of the Location header.

9. 304 ("Not Modified") should be used to preserve bandwidth.

This status code is similar to 204 ("No Content") in that the response body must be empty. The key distinction is that 204 is used when there is nothing to send in the body, whereas 304 is used when there *is* state information associated with a resource but the client already has the most recent version of the representation.

10. 307 ("Temporary Redirect") should be used to tell clients to resubmit the request to another URI.

    HTTP/1.1 introduced the 307 status code to reiterate the originally intended semantics of the 302 ("Found") status code. A 307 response indicates that the REST API is not going to process the client's request. Instead, the client should resubmit the request to the URI specified by the response message's Location header.

    A REST API can use this status code to assign a *temporary* URI to the client's requested resource. For example, a 307 response can be used to shift a client request over to another host.

11. 400 ("Bad Request") may be used to indicate nonspecific failure.

    400 is the generic client-side error status, used when no other 4xx error code is appropriate.

    For errors in the 4xx category, the response body may contain a document describing the client's error (unless the request method was HEAD).

12. 401 ("Unauthorized") must be used when there is a problem with the client's credentials.

    A 401 error response indicates that the client tried to operate on a protected resource without providing the proper authorization. It may have provided the wrong credentials or none at all.

13. 403 ("Forbidden") should be used to forbid access regardless of authorization state.

    A 403 error response indicates that the client's request is formed correctly, but the REST API refuses to honor it. A 403 response is *not* a case of insufficient client credentials; that would be 401 (“Unauthorized”).

REST APIs use 403 to enforce application-level permissions. For example, a client may be authorized to interact with some, but not all of a REST API’s resources. If the client attempts a resource interaction that is outside of its permitted scope, the REST API should respond with 403.

14. 404 (“Not Found”) must be used when a client’s URI cannot be mapped to a resource.

    The 404 error status code indicates that the REST API can’t map the client’s URI to a resource.

15. 405 (“Method Not Allowed”) must be used when the HTTP method is not supported.

    The API responds with a 405 error to indicate that the client tried to use an HTTP method that the resource does not allow. For instance, a read-only resource could support only GET and HEAD, while a controller resource might allow GET and POST, but not PUT or DELETE.

    A 405 response must include the Allow header, which lists the HTTP methods that the resource supports. For example: Allow: GET, POST.

16. 406 (“Not Acceptable”) must be used when the requested media type cannot be served.

    The 406 error response indicates that the API is not able to generate any of the client’s preferred media types, as indicated by the Accept request header. For example, a client request for data formatted as *application/xml* will receive a 406 response if the API is only willing to format data as *application/json*.

17. 409 (“Conflict”) should be used to indicate a violation of resource state.

    The 409 error response tells the client that they tried to put the REST API’s resources into an impossible or inconsistent state. For example, a REST API may return this response code when a client tries to delete a non-empty store resource.

18. 412 (“Precondition Failed”) should be used to support conditional operations.

    The 412 error response indicates that the client specified one or more preconditions in its request headers, effectively telling the REST API to carry out its request only if certain conditions were met. A 412 response indicates that those conditions were not met, so instead of carrying out the request, the API sends this status code.

19. 415 (“Unsupported Media Type”) must be used when the media type of a request’s payload cannot be processed.

    The 415 error response indicates that the API is not able to process the client’s supplied media type, as indicated by the Content-Type request header. For example, a client request including data formatted as *application/xml* will receive a 415 response if the API is only willing to process data formatted as *application/json*.

20. 500 (“Internal Server Error”) should be used to indicate API malfunction.

    500 is the generic REST API error response. Most web frameworks automatically respond with this response status code whenever they execute some request handler code that raises an exception.

Security

Authentication is delegated to the application server, so each entry point should not have to verify the authentication token. HTTP Basic authentication is configured by default through your OpenPages application server. Basic authorization schema is defined by RFC 2617, and is implemented by the client sending an HTTP request header. Basic authorization contains the text “Basic” followed by the username and password separated by colon ‘:’ and encoded in Base64. For example, if the client wishes to send the username "Aladdin" and password "open sesame", it would use the following HTTP header field:

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

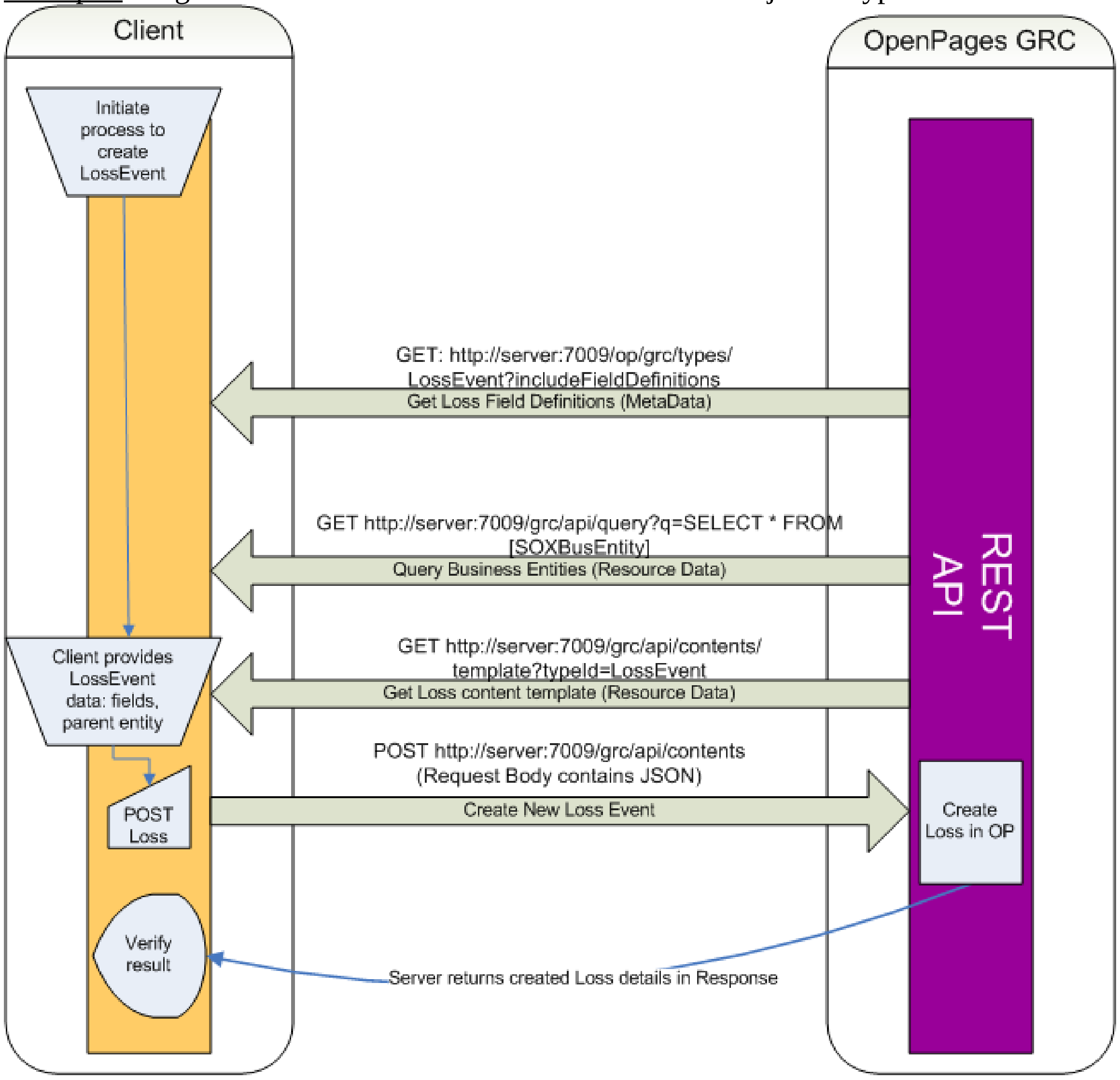
The OpenPages REST API adheres to the RESTful systems principle of statelessness, so each request made must provide the credentials of the user. Basic authentication is not considered secure unless used with with some external secure system such as SSL. For greatest security the clients should avoid sending unencrypted credentials and requests should be performed over HTTPS.

For configuring application server security, such as single signon, please refer to the IBM OpenPages GRC Platform Installation Guide and your application server's documentation.

Client-Server Interactions

Client interaction with the OpenPages REST API, including sequence of events, depends on the design of the client and the interaction objectives. Typically, an interaction will retrieve Metadata, and/or Configuration information, and then make further requests to the server to gather more data or perform updates.

Example: Diagram of the interaction to create a new GRC Object of type LossEvent


Client
OpenPages GRC
Initiate process to create LossEvent
GET: http://server:7009/op/grc/types/LossEvent?includeFieldDefinitions
Get Loss Field Definitions (MetaData)
GET http://server:7009/grc/api/query?q=SELECT * FROM [SOXBusEntity]
Query Business Entities (Resource Data)
REST API
Client provides LossEvent data: fields, parent entity
GET http://server:7009/grc/api/contents/template?typeId=LossEvent
Get Loss content template (Resource Data)
POST http://server:7009/grc/api/contents (Request Body contains JSON)
POST Loss
Create New Loss Event
Create Loss in OP
Verify result
Server returns created Loss details in Response

REST Samples

The GRC API includes samples which demonstrate client-code which uses the REST APIs. Samples are located in the grc_api/samples installation directory. The provided samples demonstrate common use cases and offer developers a starting point for developing new applications.

AnonLossEventFormREST

This demo illustrates using the Remote REST APIs to fulfill a use case for anonymous data entry. The following provides an overview of this sample:

- Allow users to enter in financial losses anonymously without logging in to the OpenPages Platform.
- Users aren’t required to be created in OpenPages security.
- Users will navigate to a web page on a company intranet or existing web portal infrastructure and fill out the form with details of the Loss.
- Upon submission, the Loss may be created in OpenPages using new REST API.

TivoliDirectoryIntegratorConnector

IBM Tivoli Directory Integrator (TDI) is an integration framework Graphical Development Environment to build and test solutions Web-based Administration and Monitoring Console for management (AMC). It has connectors for many common protocols and systems. It also has plugin architecture for making custom connectors/adapters. It allows for data transformation, mapping for data from multiple systems in a defined flow called an "Assembly Line".

This sample creates a very basic Assembly Line with an OpenPages connector written against the OpenPages REST API. The sample will pull LossEvents from OpenPages based on a Query that is written in the QueryService syntax. The LossEvents are then mapped to a database table called "Losses" that represents an external system.

Prerequisite: This sample requires IBM Tivoli Directory Integrator 7.1.1 to be installed.