REST API

Contents

Chapter 1. REST API

A REST API is an architectural style for building web applications that relies on HTTP methods. An application based on REST API communicates with the server by sending HTTP requests (such as GET, POST, PUT, DELETE) to specific URLs. The REST API provides access to IBM Z Software Asset Management Db2® repositories using common HTTP operations, enabling you to communicate with servers and exchange data.

This REST API is available only for IBM Z Software Asset Management Db2® repositories.

Prerequisites

The installation process requires the following:

- IBM Open Enterprise SDK for Python, version 3.11.
- Valid TSO userid. This userid must have Db2® authority to operate on repositories (from SYSOPR privilege, or higher).
- C/C++ Compiler (**z/OS XL C/C++** is the preferred option. If not available, use the **IBM clang compiler**).
- Register a Rule with the Policy Agent to ensure you have AT-TLS, as this is the only supported security schema. For more information see step 7 on page 5 in "Customizing and running the REST API". Contact your system administrator before modifying any page or rule as this is an OS related activity.

Note: This REST API service supports IBM Z Software Asset Management 8.3 and above.

Installing the REST API

This topic describes the steps to install REST API.

Before you begin

a. Use SMP/E to install the IBM Z Software Asset Management 8.3.1 target libraries.

b. The REST API service is delivered in the form of a PAX file and this PAX file also contain the following OSS components:

- Django-5.0.3
- asgiref-3.7.2
- djangorestframework-3.14.0
- gunicorn-21.2.0
- packaging-23.2
- python-ibmdb-3.2.3
- pytz-2024.1
- regex-2023.12.25
- sqlparse-0.5.1
- typing_extensions-4.10.0
- tzdata-2024.1

c. The pax file is in the 8.3.1 hsi.SHSIPAX1 target library.

Procedure

1. Create the API installation directory. E.g. `mkdir /api_install`
2. `cd /api_install`
3. Extract the PAX file contents and keep a note of the directory, (e.g. /api_install/dist). You can extract the PAX file contents from the hsi.SHSIPAX1 target library using the command:

   ```
   pax -rv -f "//'hsi.SHSIPAX1(HSIPAX01)'"
   ```

Customizing and running the REST API

This topic describes how to configure and run your REST API service.

About this task

The following members need to be customized:

Table 1.

JCLLLIB/PARMLIB members	Description
HSISAPIE (PARMLIB)	Environment variable.
HSISAPIP (JCLLIB)	Job to invoke the REST API.

Table 2.

UNIX member	Description
odbc.ini	This file is referenced by parameter DSNAOINI in PARMLIB member HSISAPIE. File 'odbc.ini' is provided under the API installation folder. In this file only parameter MVSDEFAULTSSID needs to be updated with the Db2 subsystem name or a Group Attach name (for DSG).

Note: In `odbc.ini` file, the field CURRENTAPPENSCH must be set to ASCII. If you get a **UnicodeDecodeError**, the reason could be that CURRENTAPPENSCH is set to EBCDIC.

Procedure

1. Customize PARMLIB member HSISAPIE.

Parameters	Description
UNX_API_PATH	Path to the API installation folder (e.g. /api_install/dist).
PROD_PORT	Port where API will listen for incoming requests (e.g. 8000).
PROD_WORKERS	Number of concurrent request handlers (recommended value of 3).

Parameters	Description
IBM_DB_HOME	The hlq for Db2® libraries used for accessing SDSNC.H and SDSNMACS. If SDSNMACS uses a different prefix, then define Db2_MACS as an environment variable. If SDSNHC.H uses a different prefix, then define Db2_INC as an environment variable.
DSNAOINI	The odbc.ini file containing ODBC/CLI parameters. A sample is provided under the API installation folder and must be customized according to your site requirements. Note: File must be tagged as either binary or IBM-1047 with text tag set to 'off'.(T=off when running chtag -p odbc.ini command).

2. Customize JCLLIB member HSISAPIP. Following is an example:

```
//HSISAPIP JOB, 'REST API job',REGION=0M,TIME=NOLIMIT,
// CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//* STEP1: Launch the REST API job
//STEPEXEC EXEC PGM=BPXBATCH,REGION=0M,TIME=NOLIMIT
//STEPLIB DD DISP=SHR,DSN=db2.SDSNEXIT
//        DD DISP=SHR,DSN=db2.SDSNLOAD
//        DD DISP=SHR,DSN=db2.SDSNLOD2
//*
//STDIN    DD PATH='/dev/null',PATHOPTS=(ORDONLY)
//STDOUT   DD SYSOUT=*
//STDERR   DD SYSOUT=*
//STDENV   DD DISP=SHR,DSN=&HSIINST.PARMLIB(HSISAPIE)
//STDPARM  DD *
SH cd $UNX_API_PATH; ./run.sh --prod
/*
```

   Replace the hlq for STEPLIB library `db2.SDSNLOD2.`

   Replace `&HSIINST` with the value specified in the customization job, HSISCUST.

3. In odbc.ini, parameter MVSDEFAULTSSID needs to be updated with the Db2® subsystem name or Group Attach name (for DSG).
   Note: File must be tagged as either binary or IBM-1047 with text tag set to 'off'. (T=off when running chtag -p odbc.ini command).
4. Update the user .profile file. To run Python you need to set the Python installation path.
   Following is an example:

```
export LIBPATH=/hcl/tcypz3b/usr/lpp/IBM/cyp/v3r11/pyz/lib:$LIBPATH
export PATH=/hcl/tcypz3b/usr/lpp/IBM/cyp/v3r11/pyz/bin:$PATH
```

5. Optionally, a subset of API configurations can be provided directly on UNIX, with the configuration script `config.sh` contained in the root project directory. This configuration option can be used to further configure UNIX environment variables, such as the Python installation PATH, or LIBPATH variables and C Compiler related options.
6. If you want to change compiler, modify the CC and CXX flags inside the `config.sh` file in the root folder of the API.
7. To enable AT-TLS, configure the policy agent (PAGENT) to match the port where the API is running with the jobname (HSISAPP). The rules can be defined in the `/etc/pagent/` folder. If you are unsure about this step, contact your system administrator.

Note: If the API is run without AT-TLS, it is only accessible using HTTP. For example, if you have AT-TLS configured on **port 6789** for jobname **HSISAPIP**, but the API is running on a different port, or with a different jobname (i.e. port **6788** for jobname **HSISAPIC**), then you can query the API on **port 6788** using HTTP. Conversely, if the API is running with a matching policy, the API will only be available using HTTPS.

Running the REST API

This topic describes how to run the REST API. The HSISAPIP job in the JCLLIB library is used to start up the REST API service.

About this task

Procedure

1. Submit HSISAPIP.
2. To query the REST API, you can use tools such as CURL, or other REST API client tools.

Endpoints

This topic describes the REST API endpoints.

Domain name and listening port

To obtain the domain and listening port, see job output of HSISAPIP as follows:

```
Domain name/Listening Port: https://pthomu1.prod.com:8000
```

Any Query

`(api/any-query/)`

This endpoint enables you to pose a single arbitrary SQL SELECT statement against the underlying repository.

To reference the repository, you need to supply the domain or IP address of this repository.

The endpoint is called using an HTTP POST request. Syntax and parameters are provided as follows:

Table 3. Parameters

Name	Type	Description	Required
query	String	SQL SELECT statement	Yes
repository	String	Name of repository	Yes

For example:

```
curl https://pthomu1.prod.com:8000/api/any-query/
-X POST
-u username:password
```

```
--form "query=SELECT * FROM table"
--form "repository=TADZREP1"
```

End of Service Products

This endpoint provides a summary of discovered products that have a known *End of Service* date. The endpoint is called using an HTTP GET request. Syntax and parameters are provided as follows:

Table 4. Parameters

Name	Type	Description	Required
vendor	String	Name of vendor	No
product_release	String	Name of product release	No
show_system_names	Boolean	Show system names	No
show_sysplex_names	Boolean	Show sysplex names	No
include_non_eos_products	Boolean	Include non-EOS products	No
start_date	String	Start date - with format YYYY-MM-DD	No
end_date	String	End date - with format YYYY-MM-DD	No
repository	String	Target repository	Yes
knowledge_base	String	Whether to use the GKB or GKU	No

For example:

```
curl https://pthomu1.prod.com:8000/api/products/end-of-service/
-X GET
-u username:password
```

Products Inventory

Table 5. Parameters

Name	Type	Description	Required
vendor	String	Product vendor	No
product	String	Product name	No
system	String	System name	No
show_product_discovery_name	Boolean	Show product discovery name	No
show_subcap_values	Boolean	Show sub cap values	No
show_sysplex_names	Boolean	Show sysplex names	No

Table 5. Parameters (continued)

Name	Type	Description	Required
show_system_names	Boolean	Show system names	No
show_product_suites_only	Boolean	Show product suites only	No
repository	String	Target repository	Yes

For example:

```
curl https://pthomu1.prod.com:8000/api/products/inventory/
-X GET
-u username:password
```

Product Use by Machine

This endpoint provides cross reference information of product versions used per Machine. The endpoint is called using an HTTP GET request. Syntax and parameters are provided as follows.

Table 6. Parameters

Name	Type	Description	Required
vendor	String	Vendor of the product	No
product_name	String	Product name	No
start_date	String	Start date - with format YYYY-MM-DD	No
end_date	String	End date - with format YYYY-MM-DD	No
show_product_discovery_names	Boolean	Show the product discovery names	No
show_service_and_support_pid	Boolean	Show the service and support pid	No
show_hardware_partition	Boolean	Show the hardware partition	No
metrics: • LAST_USE_DATE • FIRST_USE_DATE • MODULE_EVENTS • PRODUCT_USE • USER_ID_COUNT • JOB_NAME_COUNT • JOB_ACCOUNT_COUNT • SCRT_MSU	String	Default value of LAST_USE_DATE	No

Table 6. Parameters (continued)

Name	Type	Description	Required
repository	String	Target repository	Yes

For example:

```
curl https://pthomu1.prod.com:8000/api/products/usage/machine/
-X GET
-u username:password
```

Product Use by Month

This endpoint provides cross reference information of product versions used per Month by System. The endpoint is contacted via an HTTP GET request. Syntax and parameters are provided as follows:

Table 7. Parameters

Name	Type	Description	Required
vendor	String	Vendor name	No
product_name	String	Product name	No
system	String	System name	No
start_date	String	Start date - with format YYYY-MM-DD	No
end_date	String	End date - with format YYYY-MM-DD	No
show_product_discovery_name	Boolean	Show product discovery name	No
show_s_s_pid	Boolean	Show S&S PID	No
metrics: • MODULE_EVENTS • PRODUCT_USE • USER_ID_COUNT • JOB_NAME_COUNT • JOB_ACCOUNT_COUNT • SCRT_MSU	String	Default value of MODULE_EVENTS	No
repository	String	Target repository	Yes

For example:

```
curl https://pthomu1.prod.com:8000/api/products/usage/month/
 -X GET
 -u username:password
```

Repositories

This endpoint enables the management and interaction of the REST API repositories. This family of endpoints allows you to decide which of your repositories are exposed using the API.

Note: In creating and deleting repositories, you are adding an existing repository to the list of repositories that the API will recognize. As a general rule, this API will have only read access to the Db2® server.

Adding repositories

The REST API has three methods of adding repositories. The repositories are first validated by the API.

1. Add a single repository using a PUT request:

   ```
   https://pthomu1.prod.com:8000/api/repositories/<repository name>
   ```

2. Add multiple repositories at one time using a POST request:

   ```
   https://pthomu1.prod.com:8000/api/repositories/
   ```

   where the body uses the following JSON structure:

   ```
   {"repositories": ["repo1","repo2"]}
   ```

3. Add all of the available repositories using a PATCH request:

   ```
   https://pthomu1.prod.com:8000/api/repositories/
   ```

Deleting repositories

You can delete a REST API repository by calling a DELETE request:

```
https://pthomu1.prod.com:8000/api/repositories/<repository name>
```

Retrieving repository information

You can either GET a list of available repositories by querying the endpoint:

```
https://pthomu1.prod.com:8000/api/repositories/
```

or GET the repository TPARAM table contents by querying the endpoint:

```
https://pthomu1.prod.com:8000/api/repositories/<repository name>
```

Index