REST APIs for Direct Db2 Access

Tim Rowe – timmr@us.ibm.com
Architect Application Development IBM i
What is an API - Agenda

— What is an API
— What is a Web Service
— SOAP vs REST
  • What is SOAP
  • What is REST
  • Benefits
  • Drawbacks
— REST for SQL ?? YES!!
Connections

There

Devices

Applications

Here
API Definition

Application Programming Interface
Application programming interface

From Wikipedia, the free encyclopedia

"API" redirects here. For other uses, see API (disambiguation).

In computer programming, an **Application Programming Interface (API)** is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. An API may be for a web-based system, operating system, database system, computer hardware or software library. An API specification can take many forms, but often includes specifications for routines, data structures, object classes, variables or remote calls. POSIX, Microsoft Windows API, the C++ Standard Template Library and Java APIs are examples of different forms of APIs. Documentation for the API is usually provided to facilitate usage.
APIs - Simple

Simple way to connect endpoints. Send a request and receive a response.
What is an API? It’s a Web Service....what is a Web Service?

... a service?

A repeatable business task – e.g., check customer credit; open new account

Available on the Web
SOAP vs REST
Simple Object Access Protocol
What is SOAP

- Exposes **operations** that implement logic
- Designed for distributed computing
- Standardized
- Aligns with Enterprise Application needs
  - Support multi transport
  - Enterprise security – WS.Security
  - Governance with strong typing
  - Broad Development tooling support
- XML Based message protocol
- Uses WSDL as a contract between consumer and provider
REpresntational
State
Transfer
What is REST

- Architectural Style as described by Roy Fielding
- Resource focused
- Every request is via hyperlink ie http request
- Easily consumed by any client, especially web clients
- Light weight
  - Uses JSON vs XML
  - No required header for each message
- Resources are driven by HTTP Specification
  - GET, PUT, DELETE, POST
Why one vs the other? Philosophical Difference

**SOAP**
- Enterprise Driven
- Contract based
- Robust Infrastructure
- More Security Options

**Rest**
- Simplicity
- Small packet size
- HTTP focused
- Easy to call from JavaScript
SOAP vs. REST example

Is the light bulb currently on?

**SOAP**

- **Service:** LightbulbManagement(\[http://w3/services/lightbulbMgmt\])
- **Operation:** getState
  - **Input:** lightbulbID [string]
  - **Output:** state [On | Off]
- **Operation:** setState
  - **Input:** lightbulbID [string], state [On | Off]
  - **Output:** void

**REST**

- **Resource:** lightbulbs/sam
  - **Operation:** GET
    - **Output:** state [On | Off]
  - **Operation:** POST
    - **Input:** state [On | Off]
    - **Output:** nothing or resource representing the new state
SOAP vs. REST example data flows

**SOAP request**

```xml
POST /services/LightBulbManager HTTP/1.1
Host: example.com
Content-Type: text/xml; charset=UTF-8
SOAPAction: "LightBulbManager#getState"

<?xml version='1.0' ?>
<env:Envelope xmlns:env="...">
  <env:Body>
  <ns1:getState xmlns:ns1="...">
    <in0 xsi:type="xsd:string">SAM</in0>
  </ns1:Body>
</env:Envelope>
```

**REST request**

```http
GET http://w3/lightbulbs/SAM HTTP/1.1
Host: example.com
Accept: application/xml
```

POST /services/LightBulbManager HTTP/1.1
Host: example.com
Content-Type: text/xml; charset=UTF-8
SOAPAction: "LightBulbManager#getState"

<?xml version='1.0' ?>
<env:Envelope xmlns:env="...">
  <env:Body>
  <ns1:getState xmlns:ns1="...">
    <in0 xsi:type="xsd:string">SAM</in0>
  </ns1:Body>
</env:Envelope>
```
IBM i Integrated Web Services Environment

**IBM i: Integrated Web Services Server** SOAP & REST

Included with IBM i
About Integrated Web Services

— Released December of 2007 on IBM i 5.4, 6.1, and 7.1
  • Installed as part of base operating system option 3
  • Always load latest HTTP Group PTF for latest fixes and enhancements
— Consists of two separate entities
  • Integrated web services client for ILE
  • Integrated web services server
— Latest information, including product prerequisites, can be found at http://www.ibm.com/systems/i/software/iws/
— Continues to be re-invented and enhanced on 7.2 & 7.3 & 7.4
Logistics

URL – http://getSpecialReport

Http Server
- Web Server Listening for Requests

Application Server
- Process Request
- Parse the Web Stuff
- Call back end PGM

Normal IBM i Job
- RPG PGM Super Special Report

Db2 Data

SQL Statement
- JDBC Connection
Rest Apis - Today

Built in REST Apis connection for program calls

- RPG PGM / SRVPGM
- COBOL PGM / SRVPGM
- Java / C / C++
Connect to the Database - Today

From Web need to use Db connection to access DB directly

Need to connect with
- JDBC
- ODBC

RPG PGM / SRVPGM
COBOL PGM / SRVPGM
Java / C / C++

IWS Server
Db2 Server Jobs
Db2 for i
Rest APIs – New SQL Access Using REST

IWS server handles the connection to the database. Use SQL directly from a REST connection.

Use 1 connection from Web to IBM i – All HTTP based

- RPG PGM / SRVPGM
- COBOL PGM / SRVPGM
- Java / C / C++

Db2 for i

IWS server handles the connection to the database. Use SQL directly from a REST connection.
Multi – Tier

HTTP Server

IWS Server

RPG PGM / SRVPGM

COBOL PGM / SRVPGM

Db2 Server Jobs

Db2 for i
What are all the parts...

- HTTP Apache Server
  - Connector to the IAS server

- IAS Server (Liberty)
  - JAX-RS (REST)
  - JAX-WS (SOAP)

- Java program
  - Handles Inputs
  - Calls the backend ILE Program
  - Converts Output back to Web format
About integrated web services server REST support

— Uses JAX-RS
  • Java API for RESTful Web Services

— Two ways to deploy a REST service
  • IBM Web Administration GUI updated
    o Deploying a REST service will require more user input than when deploying a SOAP service
  • QShell script installWebService.sh updated to support REST
  • Qshell Scripts for virtually all functions within IWS, everything can be programmatically configured
What to get Prepared Ahead of Time

• Identify the RPG / Cobol / Java pgm
• Identify the SQL statements to use
• Figure out the HTTP methods
  • GET   - read activities
  • POST  - create entries
  • PUT   - update an entry
  • DELETE – remove
• Determine the URI - Identifiers
  • Use Nouns vs Verbs
  • Keep it simple
• What incoming content types need to be supported
• What type of data is to be returned
Student Registration Management

Create APIs using SQL to do basic application function

- Register new students
- Edit registered student information
- List registered students
- Get information about a student
- Remove student registrations

Thanks Nadir!
Getting Started

• Specify media types (e.g. XML, JSON, etc.) the procedure will accept

• Specify media types the procedure will return

• Optionally specify what values to inject in procedure input parameters
  • Path segment (e.g. /accounts/{id})
  • Matrix parameters (e.g. /cars; color=blue)
  • Query parameters (e.g. /cars?color=blue)
  • Form data
  • HTTP headers
  • HTTP Cookies

• Optionally designate response code and HTTP header output parameters
## HTTP Methods and URI Mappings

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/context-root/students</td>
<td>Return all student registrations</td>
</tr>
<tr>
<td>GET</td>
<td>/context-root/students/{id}</td>
<td>Return student registration</td>
</tr>
<tr>
<td>POST</td>
<td>/context-root/students</td>
<td>Register a new student</td>
</tr>
<tr>
<td>PUT</td>
<td>/context-root/students</td>
<td>Update registered student</td>
</tr>
<tr>
<td>DELETE</td>
<td>/context-root/students/{id}</td>
<td>Remove registered student</td>
</tr>
</tbody>
</table>
# Lets Define the SQL

<table>
<thead>
<tr>
<th>URI</th>
<th>Procedure identifier</th>
<th>SQL statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>/context-root/students</td>
<td>GETALL</td>
<td>SELECT * from STUDENTDB</td>
</tr>
<tr>
<td>/context-root/students/{id}</td>
<td>GETBYID</td>
<td>SELECT * from STUDENTDB WHERE &quot;studentID&quot; = ?</td>
</tr>
<tr>
<td>/context-root/students</td>
<td>ADD</td>
<td>INSERT INTO STUDENTDB (&quot;studentID&quot;, &quot;firstName&quot;, &quot;lastName&quot;, &quot;gender&quot;) VALUES(?,?,?,?)</td>
</tr>
<tr>
<td>/context-root/students</td>
<td>UPDATE</td>
<td>UPDATE STUDENTDB SET &quot;firstName&quot; = ?, &quot;lastName&quot; = ?, &quot;gender&quot; = ? WHERE &quot;studentID&quot; = ?</td>
</tr>
<tr>
<td>/context-root/students/{id}</td>
<td>REMOVE</td>
<td>DELETE FROM STUDENTDB WHERE &quot;studentID&quot; = ?</td>
</tr>
</tbody>
</table>
Let's Setup the Database

Open Run SQL Scripts in ACS

Create the Library

**CL: CRTLIB STUDENTRSC**

Create the Database

CREATE TABLE STUDENTRSC/STUDENTDB
("studentID" CHAR (9) NOT NULL,
 "firstName" CHAR (50) NOT NULL,
 "lastName" CHAR (50) NOT NULL,
 "gender" CHAR (10) NOT NULL,
 PRIMARY KEY ("studentID")
)

Populate the Database

INSERT INTO STUDENTRSC/STUDENTDB
("studentID", "firstName", "lastName", "gender")
VALUES('823M934LA', 'Nadir', 'Amra', 'Male'),
('826M660CF', 'John', 'Doe', 'Male'),
('747F023ZX', 'Jane', 'Amra', 'Female')
Setup

In order to insure the service works, need to authorize the user profile for the service to the Db2 table

```
CL: CHGAUT OBJ('/qsys.lib/studentrsc.lib/studentdb.file')
    USER(QWSERVICE) DTAAUT(*RWX)
```
Web Integration Permissions

In the past, any user wanting to use Web Admin they were required to have *ALLOBJ and *IOSYSCFG special authority!

System Security policy just does not allow this!

‘Permissions’ Support

- Now a *USER granted ‘permission’ can use the GUI
- Group profiles are now supported
Web Integration Permissions

— Developers can use Web Admin
  • No longer need *ALLOBJ special authority
  • Administrators can grant users ‘Permission’
  • Empowering the User
  • Group Profile support

— Two Permissions Available
  • Operator – Start & Stop servers
  • Developer – All functions

Integrated GUI interface now available to Developers and Operators without compromising your system security
Create web services server


Click on the Create New Web Services Server link
How do you test things?

Free Download  https://www.soapui.org
Deployment demo

Notes:

1. This panel has been updated so one can indicate whether web service is based on SQL statements or ILE programs/service programs
Create the Web Services Server Container

Create Web Services Server

Specify Web services server name - Step 1 of 4

Welcome to the Create Web Services Server wizard. A Web services server provides a secure and easy way to configure the environment for hosting Web services that are based on IBM i objects such as RPG and COBOL programs and SQL statements. The wizard creates everything needed to run Web services.

For more information, please visit: http://www.ibm.com/support/docview.wss?uid=isg3T1026868

Specify a unique name for this server

Server name: Virtual

Server description: Web Services Server for Common Virtual Conference 2019

Create HTTP server
Create the Web Services Server Container

Create Web Services Server

Specify network attributes for server - Step 2 of 4

Your server may listen for requests on specific IP addresses or on all IP addresses of the system. A common server.

Specify internet addresses and ports for server

Specify server command port: 10259

Specify internet address and port for the server

IP address: All IP addresses
Port: 10258

Specify internet address and port for the HTTP server

IP address: All IP addresses
Port: 10268

Back  Next  Cancel
Create the Web Services Server Container

Create Web Services Server

Specify User ID for Server - Step 3 of 4

The server requires an IBM i user ID to run the server's jobs. It is recommended that a special user ID is used for jobs since this user ID is given authority to all of the server's objects, such as files and directories.

Specify user ID for this server: 

- Use default user ID
  
  Note: The default server user ID is QWSERVICE.

- Specify an existing user ID
- Create a new user ID

Back  Next  Cancel
Create the Web Services Server Container

Create Web Services Server

Summary - Step 4 of 4

Web Services Server Information

Server name: Virtual
Server description: Web services server created by the Create Web Services Server wizard.
Port: 10258
Command port: 10259
Server root: /www/Virtual
Server URL: http://common1.frakeni.com:10268
User ID for server: QWSERVICE
Context root: /web

HTTP Server Information

HTTP server name: VIRTUAL
HTTP server description: Web services server created by the Create Web Services Server wizard.
Port: 10268
Document root: /www/Virtual/htdocs
Server root: /www/Virtual
Server association: Virtual
Create the Web Services Server Container

**Manage Web Services Server**

*Server: Virtual*

Web services server created by the Create Web Services Server wizard.

The IBM integrated Web services server provides a secure and easy way to configure an environment for hosting Web objects such as RPG and COBOL programs and Db2 files. An easy to use interface for managing the server in addition to services is provided.

For more information, please visit: [http://www.ibm.com/support/docview.wss?uid=isg3T1026868](http://www.ibm.com/support/docview.wss?uid=isg3T1026868)

Server "Virtual" is in the process of being created. To update the status, click the **Refresh** icon above.

**Note:** To update the status, click **Refresh**
Container Created

Manage Web Services Server

Server: Virtual

Web services server created by the Create Web Services Server wizard.

The IBM integrated Web services server provides a secure and easy way to configure an environment for Web services that are based on IBM i objects such as RPG and COBOL programs and Db2 files. An example of managing the server in addition to installing and managing Web services is provided.

For more information, please visit: [http://www.ibm.com/support/docview.wss?uid=isg3T1026868](http://www.ibm.com/support/docview.wss?uid=isg3T1026868)

Note: To update the status, click Refresh
Deploy a SQL Based Service

Virtual > Manage Deployed Services

Manage Deployed Services

Data current as of Jul 15, 2019 7:08:17 AM.

Deployed services:

<table>
<thead>
<tr>
<th>Service name</th>
<th>Status</th>
<th>Type</th>
<th>Startup type</th>
<th>Service definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertTemp</td>
<td>Running</td>
<td>SOAP</td>
<td>Automatic</td>
<td>View WSDL</td>
</tr>
</tbody>
</table>

Deploy

Refresh

Close
Deploy a SQL Based Service

Virtual > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Web service type - Step 1 of 9

Welcome to the Deploy New Service wizard. This wizard helps you create Web services using IBM i objects and data. A Web service is a self-contained software component with a well-defined interface that describes a set of operations accessible over the Internet and exchange XML messages that are based on the SOAP protocol. A REST-based Web service exposes resources, where client requests are handled by resource methods and the format of messages that are exchanged is defined by the resource itself.

Specify Web service type:

Specify Web service implementation:

REST

*PGM/*SRVPGM

*SQL
Deploy a SQL Based Service

Specify a meaningful resource name, this service will be referenced now as:

/context-root/students
Deploy a SQL Based Service

Virtual > Manage Deployed Services > Deploy New Service

Deploy New Service
Specify Database Properties - Step 3 of 8

Specify database properties that will be used to process SQL statements.

Database system: localhost
Default schema: studentrscl
Naming convention: *SQL
Library list: *LIBL

Is the database Local or Remote? I.e., the database can be on a different IBM i.

Specify the default schema for these Rest APIs. Only this Schema is used by default.

Do you want to use SQL based naming? schema.table
Do you want to use System based naming? schema/table

Specify the library list that is to be used to resolve unqualified stores procedure names or unqualified names.
### Deploy a SQL Based Service - Specify the SQL

Specify SQL statements that will be externalized as a Web service:

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>SQL statement/Parameter name</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOVE</td>
<td>Delete From STUDENTDB Where &quot;studentID&quot; = ?</td>
</tr>
</tbody>
</table>

Enter the Procedure name and the SQL statement. Notice, I have a parameter specified.

Once you click ‘Continue’ the wizard will detect the parameter and you will have the opportunity to then view and modify.
Deploy a SQL Based Service - Specify the SQL

### Deploy New Service
Specify SQL Statements - Step 4 of 8

Specify SQL statements that will be externalized as a Web service:

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>SQL statement/Parameter name</th>
<th>Usage</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOVE</td>
<td>Delete From STUDENTDB Where &quot;studentID&quot; = ?</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td>PARM00001</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Click on the entry to ‘open’ it back up for update to modify the parameter name from the default value to something customized.
Deploy a SQL Based Service - Specify the SQL

Specify SQL statements that will be externalized as a Web service:

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>SQL statement/Parameter name</th>
<th>Usage</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOVE</td>
<td>Delete From STUDENTDB Where &quot;studentID&quot; = ?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Updated to correct value for "studentID".
## Deploy a SQL Based Service - Specify the SQL

Specify SQL statements that will be externalized as a Web service.

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>SQL statement/Parameter name</th>
<th>Usage</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOVE</td>
<td>Delete From STUDENTDB Where &quot;studentID&quot; = ?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>studentID</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td>UPDATE</td>
<td>UPDATE STUDENTDB SET &quot;firstName&quot; = ?, &quot;lastName&quot; = ?, &quot;gender&quot; = ? WHERE &quot;studentID&quot; = ?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PARM00001</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00002</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00003</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00004</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td>ADD</td>
<td>INSERT INTO STUDENTDB (&quot;studentID&quot;, &quot;firstName&quot;, &quot;lastName&quot;, &quot;gender&quot;) VALUES(? ,?,?,?)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PARM00001</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00002</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00003</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td></td>
<td>PARM00004</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td>GETBYID</td>
<td>SELECT * from STUDENTDB WHERE &quot;studentID&quot; = ?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>studentID</td>
<td>input</td>
<td>CHAR</td>
</tr>
<tr>
<td>GETALL</td>
<td>SELECT * from STUDENTDB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Deploy a SQL Based Service – SQL Information

Deploy New Service
Specify SQL Information - Step 5 of 8

Customize how each procedure will process SQL statements. For query statements, this includes the type of result sets that may be returned and how pagination is handled for result sets.

- **Procedure name:** REMOVE
  **SQL Statement:** Delete From STUDENTDB Where "studentID" = ?

- **SQL result type:** Multi-row result set
  **Trim mode for output fields:** Trailing

- **SQL state information in response:** On errors
  **Treat warnings as SQL Errors:** Yes

- **User-defined error message:**

- **HTTP status code on SQL success:** 204 or...
  **HTTP status code on SQL failure:** 500 or...

---

Set any SQL warnings as an error condition

The SQL code 204 (no content), if that code is produced, then treat as a success
Deploy a SQL Based Service – SQL Information

**Deploy New Service**

Specify SQL Information - Step 5 of 8

Customize how each procedure will process SQL statements. For query statements, this includes the type of pagination is handled for result sets.

- Procedure name: UPDATE
- SQL Statement: UPDATE STUDENTDB SET "firstName" = ?, "lastName" = ?, "gender" = ? WHERE "studentID" = ?
- SQL result type: Multi-row result set
- Trim mode for output fields: Trailing
- SQL state information in response: On errors
- Treat warnings as SQL Errors: Yes
- User-defined error message:

The SQL code 204 (no content), if that code is produced, then treat as a success.
The SQL code 201 (Created), if that code is produced, then treat as a success.
Deploy a SQL Based Service – SQL Information

- Trim trailing blanks. This improves performance and only returns the actual data.
- Return only a single entry. This will ensure the results are not an array of objects.
- The SQL code 200 (OK), if that code is produced, then treat as a success.

HTTP status code on SQL success: 200
HTTP status code on SQL failure: 500
Deploy a SQL Based Service – Resource Information

Specify Resource Method Information - Step 6 of 8

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client request. HTTP request method to a resource method:

Procedure name: REMOVE
URI path template for resource: /
HTTP request method: DELETE
URI path template for method: /id or...
HTTP header information:
Allowed input media types: *ALL
Returned output media types: *JSON
Whether to wrap input parameters:
- Wrap input parameters
- Do not wrap input parameters

Input parameter mappings:

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Data type</th>
<th>Input source</th>
<th>Identifier</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>studentID</td>
<td>CHAR</td>
<td>*PATH_PARAM</td>
<td>id</td>
<td>*NONE</td>
</tr>
</tbody>
</table>

Specify the HTTP Method

URI path template, since we have to pass in a parameter, need to define that

Return only JSON

Specify the input source as a Path Parameter
Deploy a SQL Based Service – Resource Information

Specify Resource Method Information - Step 6 of 8

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client HTTP request method to a resource method.

Specify the HTTP Method

Send and Return only JSON

Since the request will be in the payload of the client request, we specify that the parameters should be wrapped.
Deploy a SQL Based Service – Resource Information

Specify Resource Method Information - Step 6 of 8

Procedures are mapped to resource methods. Each resource method needs to be defined to have an HTTP request method to a resource method.

Specify the HTTP Method

Specify the HTTP Method

Send and Return only JSON

Since the request will be in the payload of the client request, we specify that the parameters should be wrapped.

Specify the HTTP Method

Send and Return only JSON

Since the request will be in the payload of the client request, we specify that the parameters should be wrapped.
Deploy a SQL Based Service – Resource Information

**Deploy New Service**

*Specify Resource Method Information - Step 6 of 8*

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client requests. HTTP request method to a resource method:

<table>
<thead>
<tr>
<th>Procedure name:</th>
<th>GETBYID</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI path template for resource: /</td>
<td>GET 1</td>
</tr>
<tr>
<td>HTTP request method:</td>
<td>GET 1</td>
</tr>
<tr>
<td>URI path template for method:</td>
<td>/id 2</td>
</tr>
<tr>
<td>HTTP header template information:</td>
<td></td>
</tr>
<tr>
<td>Allowed input media types:</td>
<td>*ALL 3</td>
</tr>
<tr>
<td>Returned output media types:</td>
<td>*JSON 4</td>
</tr>
<tr>
<td>Whether to wrap input parameters:</td>
<td>Wrap input parameters 5</td>
</tr>
<tr>
<td>Input parameter mappings:</td>
<td></td>
</tr>
</tbody>
</table>

**Specify the HTTP Method**

URI path template, since we have to pass in a parameter, need to define that:

Send and Return only JSON

Specify the input source as a Path Parameter
Deploy a SQL Based Service – Resource Information

Specify Resource Method Information - Step 6 of 8

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client HTTP request method to a resource method.

- **Procedure name:** GETALL
- **URI path template for resource:** /
- **HTTP request method:** GET
- **URI path template for method:** *NONE
- **HTTP header information:**
- **Allowed input media types:** *ALL
- **Returned output media types:** *JSON

Specify the HTTP Method

Default Input of *ALL is fine, and Return only JSON
Deploy a SQL Based Service – User Profile

Deploy New Service
Specify User ID for this Service - Step 7 of 8

The service requires an IBM i user ID to run the Web service business logic. The user ID must have the necessary resources that the Web service requires.

Specify User ID for this Service: ⚫

- Use server's user ID
- Specify an existing user ID
- Use authenticated user ID

User Profile for the Service. Can be different than the profile for the server. This profile must have the necessary authority to access the data in the database in order to function correctly.
Deploy a SQL Based Service – Create Service

Deploy New Service
Summary - Step 8 of 8

When you click Finish the web service is deployed.

Resource name: students
Resource description: SQL
Service install path: /www/wservice/webservices/services/students
URI path template: /
User ID for service: *SERVER (QWSERVICE)
Service Created and Running

Manage Deployed Services

Data current as of Jul 15, 2019 11:13:31 AM.

Deployed services:

<table>
<thead>
<tr>
<th>Service name</th>
<th>Status</th>
<th>Type</th>
<th>Startup type</th>
<th>Service definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertTemp</td>
<td>Running</td>
<td>SOAP</td>
<td>Automatic</td>
<td>View WSDL</td>
</tr>
<tr>
<td>students</td>
<td>Running</td>
<td>REST</td>
<td>Automatic</td>
<td>View Swagger</td>
</tr>
</tbody>
</table>

Properties to change JDBC setting, User Profile, Edit Swagger Doc, and more

Updates to the service can be made using Re-deploy

View the Swagger document, this describes the details of the service
Test the service

The properties page and the Swagger doc have the URL to call. The default with no parameters specified returns all entries in the database table.
Test the service

Create the input data source for the new student

Add a new entry to the database
Test the service

The Multiple new entries added to the database
Demo of REST based Methodology
What have we done lately....

— 3 node support
  • HTTP on one node
  • Application Server on a node
  • Backend RPG on a node
— Use Authenticated User
— Services re-deploy
— Connection pool pre-initialization
— Variable length fields

— Many other updates as requested by the community

IWS Multi Node Server Support

Multi nodes
- Apache, IWS, and backend programs ALL on different IBM i nodes

**Diagram:**

- Apache Server (IBM i)
- IWS Server (IBM i)
- Db2 Data (IBM i)
- RPG / COBOL Programs (IBM i)

**HTTP Request**
Some Additional Light Reading

Developer Works – 3 Part Series on Rest for IBM i

Questions and Answers
## For More Information:

### Some Links You Need

<table>
<thead>
<tr>
<th><strong>IBM i Home Page:</strong></th>
<th><strong>Twitter</strong></th>
<th><strong>#Hashtags</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.ibm.com/systems/i">www.ibm.com/systems/i</a></td>
<td>@IBMSystems</td>
<td>#IBMi30</td>
</tr>
<tr>
<td></td>
<td>@COMMONug</td>
<td>#PowerSystems</td>
</tr>
<tr>
<td></td>
<td>@IBMChampions</td>
<td>#IBMi</td>
</tr>
<tr>
<td></td>
<td>@IBMSystemsISVs</td>
<td>#IBMAIX</td>
</tr>
<tr>
<td></td>
<td>@IBMiMag</td>
<td>#POWER8</td>
</tr>
<tr>
<td></td>
<td>@ITJungleNews</td>
<td>#LinuxonPower</td>
</tr>
<tr>
<td></td>
<td>@SAPonIBMi</td>
<td>#HANAIonPower</td>
</tr>
<tr>
<td></td>
<td>@SiDforIBMi</td>
<td>#ITinfrastructure</td>
</tr>
</tbody>
</table>

**IBM 30th Anniversary**  
http://ibmi30.mybluemix.net/  

**IBM Systems Magazine IBM i Edition:**  
http://ibmsystemsmag.com/ibmi/  

**Support Life Cycle:**  
https://www-01.ibm.com/software/support/ibmi/lifecycle/  

**License Topics:**  
https://www-01.ibm.com/support/docview.wss?uid=nas8N1022087  

#IBM30  
#PowerSystems  
#IBMi  
#IBMAIX  
#POWER8  
#LinuxonPower  
#OpenPOWER  
#HANAIonPower  
#ITinfrastructure  
#OpenSource  
#HybridCloud  
#BigData
For More Information:

<table>
<thead>
<tr>
<th>Blogs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IBM Blogs:</strong></td>
<td><strong>IBM Champion's Blogs:</strong></td>
</tr>
<tr>
<td>IBM Systems Magazine You and i (Steve Will)</td>
<td>IBM Systems Magazine: iDevelop (Jon Paris and Susan Gantner)</td>
</tr>
<tr>
<td>IBM Systems Magazine i-Can (Dawn May)</td>
<td>IBM Systems Magazine: iTalk with Tuoy</td>
</tr>
<tr>
<td>IBM Systems Magazine: Open your i (Jesse Gorzinski)</td>
<td></td>
</tr>
<tr>
<td>IBM DB2 for i (Mike Cain)</td>
<td></td>
</tr>
<tr>
<td>IBM DB2 Web Query for i (Doug Mack)</td>
<td></td>
</tr>
</tbody>
</table>

- [http://ibmsystemsmag.com/blogs/i-can](http://ibmsystemsmag.com/blogs/i-can)
- [http://ibmsystemsmag.com/blogs/open-your-i](http://ibmsystemsmag.com/blogs/open-your-i)
- [http://db2fori.blogspot.co.uk](http://db2fori.blogspot.co.uk)
- [http://db2webqueryi.blogspot.co.uk](http://db2webqueryi.blogspot.co.uk)
- [http://ibmsystemsmag.com/blogs/idevelop](http://ibmsystemsmag.com/blogs/idevelop)
Notices and disclaimers

© 2019 International Business Machines Corporation. No part of this document may be reproduced or transmitted in any form without written permission from IBM.

U.S. Government Users Restricted Rights — use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. This document is distributed “as is” without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity. IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.”

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer’s responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer’s business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.
Notices and disclaimers continued

— Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM’s products. IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.

— The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

— IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml