CREATORCON®

May 7–10, 2018 • Las Vegas, NV

Taking Import Sets to the Next Level with Scripted REST APIs

Toney Vecchio
Senior Developer
IHG

Shubhendu Singh
ServiceNow Developer
IHG

Take the code home by visiting: https://github.com/ToneyTime/ServiceStartsNow-ScriptedREST

© 2018 ServiceNow All Rights Reserved

servicenow™

Agenda

Review: Import Sets, Transform Maps, Import Set API

Challenge: Complex Data Structure and Preprocessing

Solution: Import Set Supported by Scripted REST API

Code Walkthrough

Questions

© 2018 ServiceNow All Rights Reserved

Speaker Introduction

NAME: Toney Vecchio

TITLE: Sr. ServiceNow Engineer

LOCATION: Atlanta, GA

FUNCTION: Global Services and Support

COMPANY: IHG—InterContinental Hotels Group

EXPERIENCE: 5 years ServiceNow Admin/Developer

EXPERTISE: Speaking Business, Thinking Solutions, Searching StackOverflow

ACHIEVEMENTS: 2 Deployments, 3 Project/Demand go-lives, multiple REST Integrations and one-man army performing road mapping, requirements gathering, coding, deployments, and training

CURRENT PROJECTS: Kingston Upgrade and Service Portal

© 2018 ServiceNow All Rights Reserved

My Company

NAME: IHG—InterContinental Hotels Group

INDUSTRY: Hospitality

MARKET FOCUS:

- 425,000 User Accounts | 2,000 Fulfillers
- 5,000 Hotels ~ 750,000 Rooms
- 100 Countries

© 2018 ServiceNow All Rights Reserved

Our Goals and Challenges

We need timely and full user core data.

- ✓ Real-time REST instead of daily excel flat file of 400k+ records
- ✓ Account for users with multiple locations
- ✓ More data!

- ✗ Complex data structure
- ✗ Two external systems: Core + supplementary
- ✗ Improved maturity in error handling

© 2018 ServiceNow All Rights Reserved

That’s Not How Bob Sees the Challenge

Bob is a new IHG Manager.

- ✘ Couldn’t get access on his first day
- ✘ IT knew nothing about him
- ✘ Cost center unknown
- ✘ Unable to access secondary locations

© 2018 ServiceNow All Rights Reserved

Let’s Build Up: Starting with Excel Import

	empID	firstName	businessTitle	location
Source A	123	Bob	Senior Manager	US430

Import Set Row	empID	firstName	businessTitle	location
u_user_import	123	Bob	Senior Manager	US430

Transform Map

SN Table	employee_number	first_name	title	location
sys_users	123	Bob	Senior Manager	US430

Import Set

Transform Map

ServiceNow Table

© 2018 ServiceNow All Rights Reserved

Import Sets Are Very Useful

✓ **Pros:**

Native Tool

Field Type Management

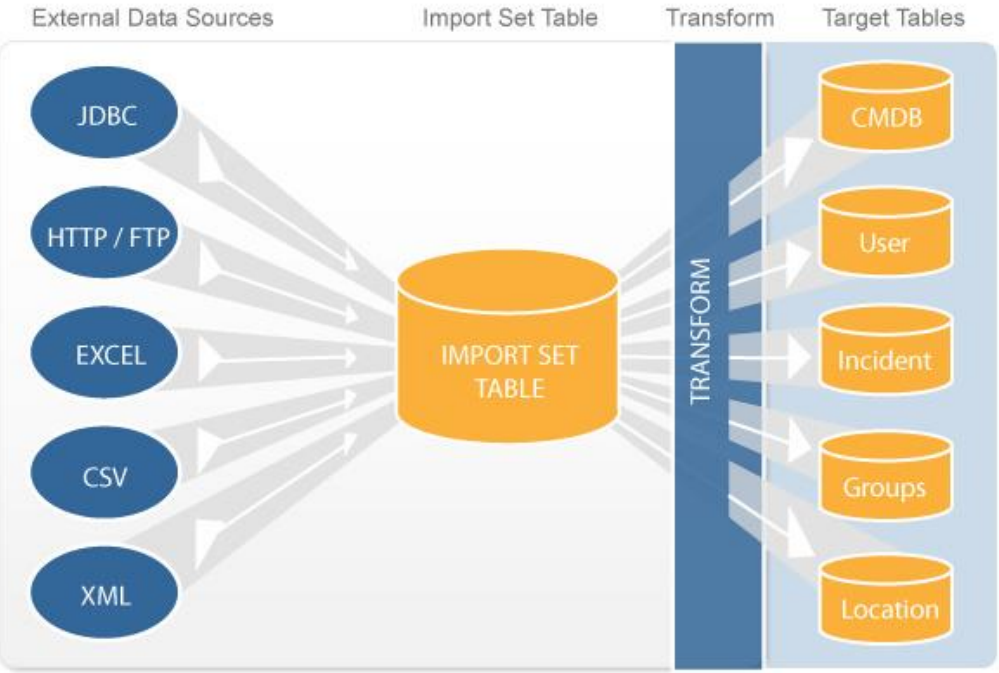
REST Enabled

✗ **Cons:**

Flat, single row inputs

Static Field Names

Error handling can be challenging



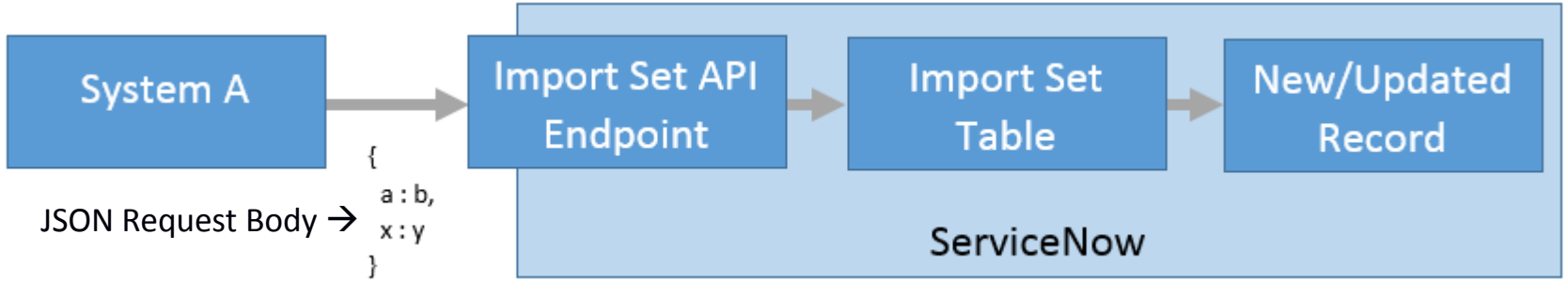
© 2018 ServiceNow All Rights Reserved

Let's Review—Import Set API and JSON

- REST API Endpoint for an Import Set Table
- **J**ava**S**cript **O**bject **N**otation used in our solution

- Example
  - **POST** https://dev1234.service-now.com /api/now/import/x_8488_ssn_rest_import_user


```
{
  "empID":"123",
  "firstName":"Bob",
  "businessTitle":"Senior Manager",
  "location":"US430"
}
```

© 2018 ServiceNow All Rights Reserved

Reality Quickly Got Tough—Complex Data

- Useful but redundant info in single message
- Must prepare if only 1 source provided

	Employee ID	First Name	Title	Location
Source A	*empID*	*firstName*	*profile.jobTitle*	*location*
	123	Bob	Head Honcho	US430
Source B	*EmployeeNum*	*f_name*	-	*locations (+)*
	123	Robert		US430, UK890, US999

User Profile Import Set Row

Multiple Locations Import Set Row

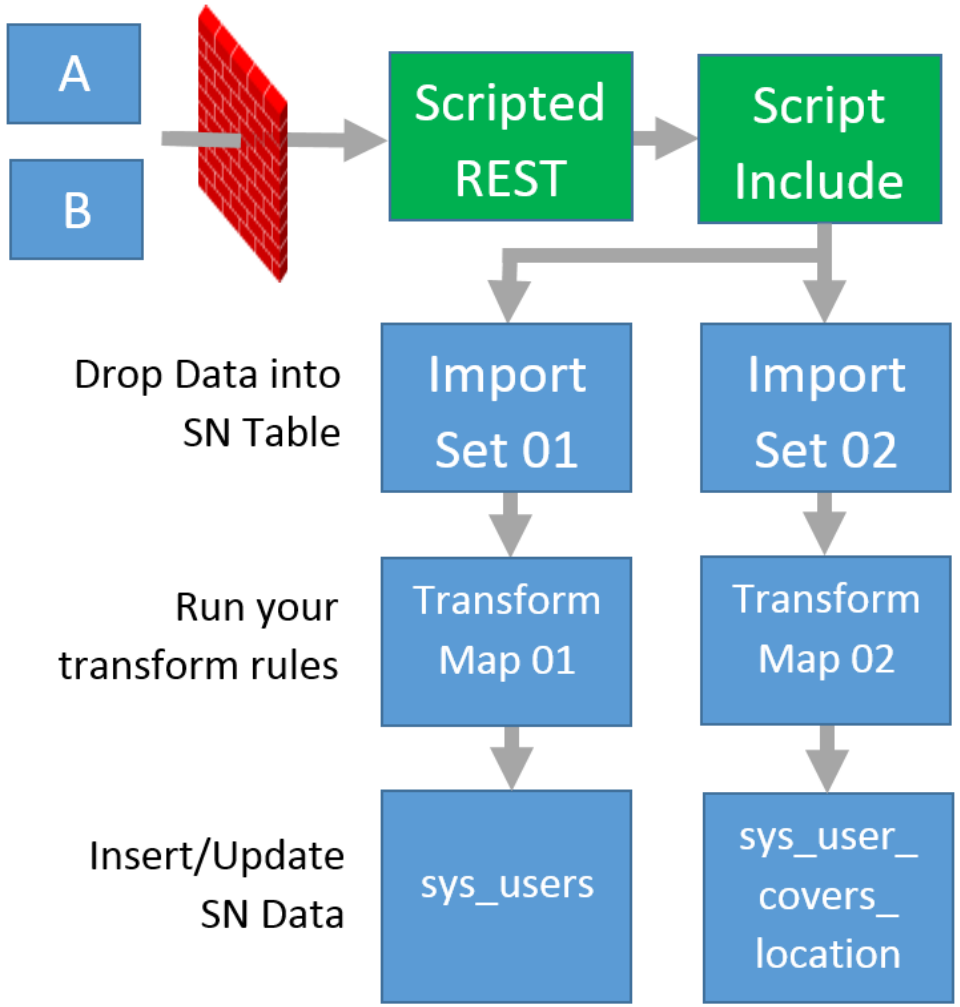

```
{
  "sourceA":{
    "employeeNum":"123",
    "firstName":"Bob",
    "profile":{
      "jobTitle":"Head Honcho",
      "businessTitle":"Senior Manager"
    },
    "location":"US430"
  },
  "sourceB":{
    "empID":"123",
    "f_name":"Robert",
    "locations":[
      {
        "primary":true,
        "loc":"US430"
      },
      {
        "primary":false,
        "loc":"UK890"
```


Source A gives me some useful data that B does not

Source B gives me multiple locations

© 2018 ServiceNow All Rights Reserved

Increased Flexibility with Scripted REST API



Adding Scripted REST lets you:

- Catch JSON before putting it into Import Set Row
- Flatten out JSON Nodes
- Manipulate field names
- Custom data integrity checks with flexible error handling
- Optional: Split content into multiple import Sets

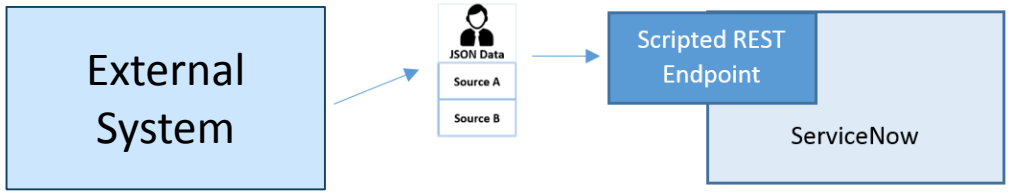
© 2018 ServiceNow All Rights Reserved

Demo

Let’s see it in action

© 2018 ServiceNow All Rights Reserved

Code Walkthrough—Source Sends JSON to Scripted REST Endpoint




```
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    gs.info('Starting Scripted REST Endpoint Processing');

    // request passed in above will be the full message sent to us
    // We only want the content from the body for now
    var j = request.body.data;

    // Because we are receiving JSON formatted data, lets print it out to the logs
    // For readability, you may need to JSON.parse(j) first, then stringify it
    gs.debug('Our Enpoint Received Body Data: ' + JSON.stringify(j));

    // Now we want to do something with the message. Lets pass the message body it into a script include.
    // We will look inside the script include, but ultimately it does its magic and sends us back an OK message or an Error Message which we will save to statusReturned variable.
    var foo = new x_8488_ssn_rest.importUtils();
    var statusReturned = foo.postUser(j);

    gs.info('Completing Scripted REST Endpoint Processing');
    gs.debug('REST Endpoint completed. Returned: ' + statusReturned.http_status);

    // Now, we have to build a response back to the system. It gave us data, we should be good netizens and respond back with a message telling them if it worked right or not.
    response.setContentType('application/json'); // Tells the sending system what form our response will be formatted in
    response.setStatus(statusReturned.http_status); // We are figuring this out in the script include
    var writer = response.getStreamWriter();
    writer.writeString(JSON.stringify(statusReturned));

})(request, response);
```

Pull the content out of the received message

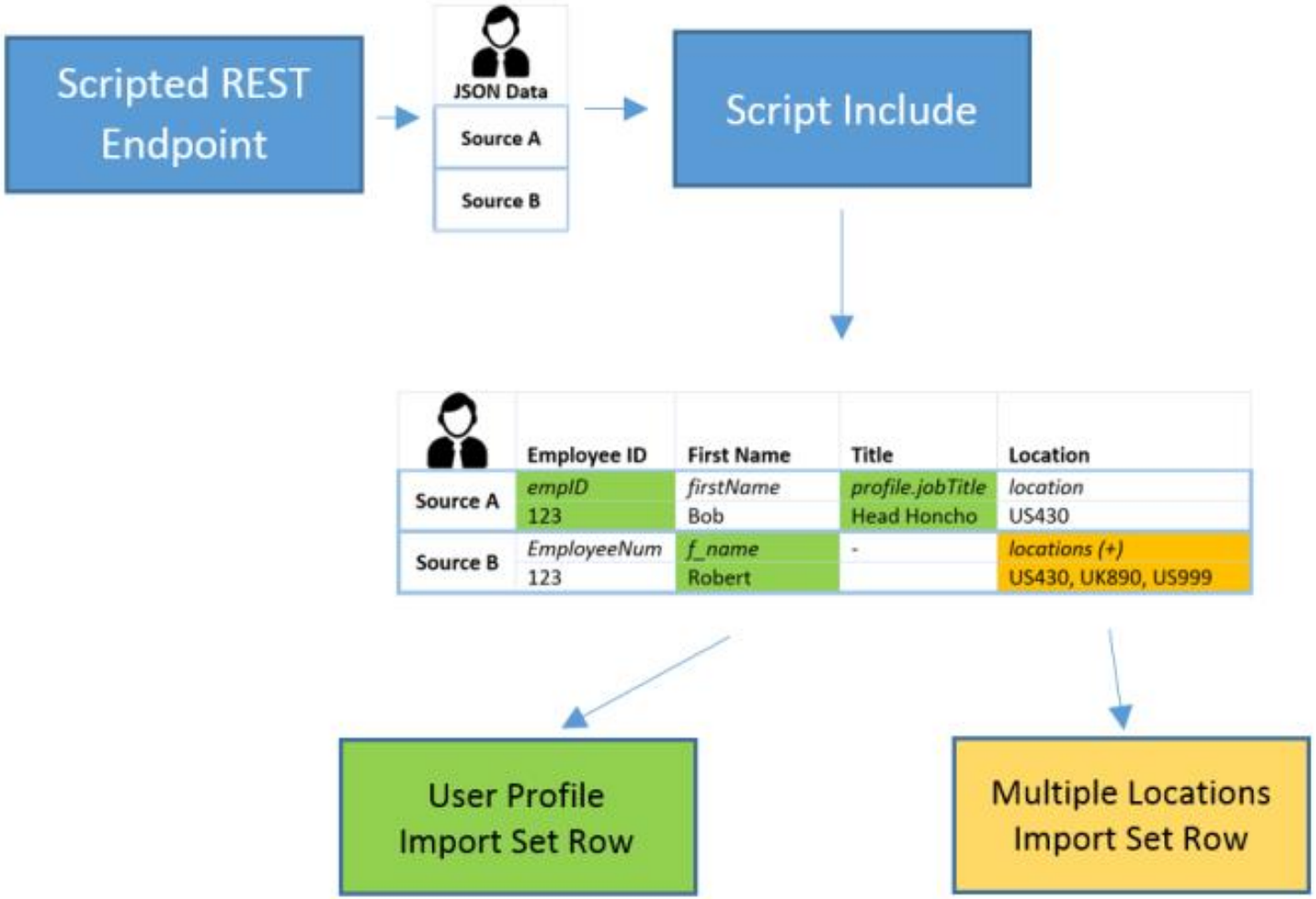
Send it to a Script Include to return an error/OK message

Respond to the source system with the custom message

© 2018 ServiceNow All Rights Reserved

REST Endpoint → Script Include → Import Set → Transform

Code Walkthrough—Script Include Prepares Our Data



© 2018 ServiceNow All Rights Reserved

Code Walkthrough—Script Include Prepares Our Data

```
var importUtils = Class.create();
importUtils.prototype = {
    initialize: function() {
    },

    postUser: function(j) {
        gs.debug('Script Include importUtils postUser has been called');

        try {
            var answer = {}; // Prepare response payload. We will write success/error messages here later
            var importSetRowUser = 'x_8488_ssn_rest_import_user';
            var importSetRowMultiLoc = 'x_8488_ssn_rest_impor_multiple_locations';

            gs.debug('Script Include received payload: ' + JSON.stringify(j)); // Lets just take a quick
look at the data

            // I like to setup the node names in advance, easier to change in the future if needed
            var nA = 'sourceA';
            var nAProfile = 'profile';
            var nB = 'sourceB';
            var nBLoc = 'locations';
```

Prepare a JSON payload to send back our custom responses to the Scripted REST endpoint

Use variables for JSON nodes and table names for easier coding and future modifications

© 2018 ServiceNow All Rights Reserved

REST Endpoint | Script Include | Import Set | Transform

Code Walkthrough—Script Include Prepares Our Data

```
23          //  -- Check Validity of JSON Content before proceding --
24          //  Looking for Employee ID in the nodes
25
26          // No Usable Data sets
27          if (j[nA] == undefined && j[nB] == undefined) {
28              answer.http_status = "400";
29              answer.status_message = 'We expected to see nodes for ' + nA + ' or ' + nB + ' but those
    were not found. Please confirm before resending';
30              gs.error('ESB Rejected: ' + JSON.stringify(answer));
31              return answer;
32          }
33
34          if (j[nA] != undefined && (j[nA].employeeNum == undefined || j[nA].employeeNum == '')) {
35              answer.http_status = "400";
36              answer.status_message = 'Received node ' + nA + ' but did not find a valid employee number.
    Please confirm before resending';
37              gs.error('ESB Rejected: ' + JSON.stringify(answer));
38              return answer;
39          }
40
41
42          gs.debug('Script Include importUtils is creating GlideRecord to insert onto user Import Set
    Row');
```

Ensure the data is structured how you expect it to be, or else warn the source system with a clear error message

Check for important fields and ensure its scrubbed and up to expectations

© 2018 ServiceNow All Rights Reserved

Code Walkthrough—Script Include Inserts to Import Set Row

```
// ----------------------------
// Prepare new records on import set row tables to receive and process customer profile
// ----------------------------

var userIn = new GlideRecord(importSetRowUser);
userIn.initialize();


// Prefer SourceA
if(j[nA] != undefined){  //  Ensure this node exists before trying to read fields
    userIn.unique_identifier = j[nA].employeeNum;
    userIn.first_name = j[nA].firstName; // Will be later overwritten if Source B is provided
    userIn.location = j[nA].location;

    // Checking to make sure node and name value pair are present
    if(j[nA][nAProfile].jobTitle != undefined){
        userIn.title = j[nA][nAProfile].jobTitle;
    }
}

// Prefer SourceB - Overriding and taking needed fields
if(j[nB] != undefined){
    userIn.first_name = j[nB].f_name;
}

// Use Case: Source A not provided. Use Source B to populate what Source A was expected to populate.
if(j[nA] == undefined && j[nB] != undefined) {
    userIn.unique_identifier = j[nB].empID;
}


// Send the Profile Data to the User Import Set Row Table
userIn.insert();
```

Prepare a record on the Import Set and start dropping desired data into it

© 2018 ServiceNow All Rights Reserved

REST Endpoint → Script Include → Import Set → Transform

Code Walkthrough—Script Include Inserts Import Set Row

```
            // Now we lookup the record that was just created moments ago so we can check how the Transform Map went.
            var importSetRow = new GlideRecord(importSetRowUser);
            importSetRow.get(resultSysID);

            // We will look to see what happened during transform and write the state of the import down
            var importSetRowStatus = importSetRow.sys_import_state;

            //  Package up all clear response
            if(importSetRowStatus == 'ignored' || importSetRowStatus == 'ignore'){
                answer.http_status = "200";
                answer.status_message = "User payload accepted and processed. Data provided matched local data set, update was ignored for " + userIn.unique_identifier;
            }

            else if(importSetRowStatus == 'updated'){
                answer.http_status = "200";
                answer.status_message = "User payload accepted and processed. Account updated for " + userIn.unique_identifier;
            }

            else if(importSetRowStatus == 'inserted'){
                answer.http_status = "201";
                answer.status_message = "User payload accepted and processed. Profile created for " + userIn.unique_identifier;

            } else {
                var msg = 'Unexpected import result during user transform: ' + importSetRow.sys_import_state.toString() + ' ' + importSetRow.sys_row_error.error_message.toString() + ' and error code is ' + importSetRow.sys_row_error.error_code.toString() + ' UserId of user ' + importSetRow.unique_identifier;
                answer.http_status = "500";
                answer.status_message = msg;

            }

            gs.debug('Script Include completing with: ' + JSON.stringify(answer));
            return answer;
```

We've put the data into an Import Set and the transform map has run

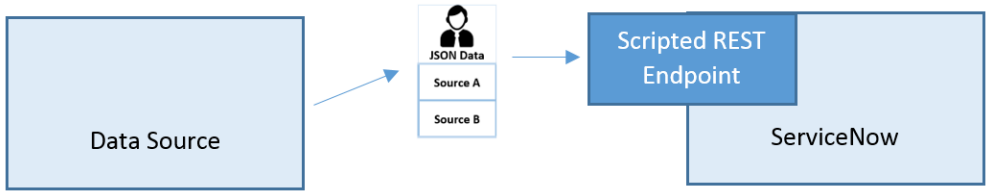
Let's fetch the results of that process and package up a response to the data source

Script Include will send back the HTTP status and status message as JSON object, allowing us to send lots of data back in one variable

© 2018 ServiceNow All Rights Reserved

REST Endpoint → Script Include → Import Set → Transform

Code Walkthrough—ServiceNow Responds to Source




```
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    gs.info('Starting Scripted REST Endpoint Processing');

    // request passed in above will be the full message sent to us
    // We only want the content from the body for now
    var j = request.body.data;

    // Because we are receiving JSON formatted data, lets print it out to the logs
    // For readability, you may need to JSON.parse(j) first, then stringify it
    gs.debug('Our Enpoint Received Body Data: ' + JSON.stringify(j));

    // Now we want to do something with the message. Lets pass the message body it into a script include.
    // We will look inside the script include, but ultimately it does its magic and sends us back an OK message or an Error Message which we will save to statusReturned variable.
    var foo = new x_8488_ssn_rest.importUtils();
    var statusReturned = foo.postUser(j);

    gs.info('Completing Scripted REST Endpoint Processing');
    gs.debug('REST Endpoint completed. Returned: ' + statusReturned.http_status);

    // Now, we have to build a response back to the system. It gave us data, we should be good netizens and respond back with a message telling them if it worked right or not.
    response.setContentType('application/json'); // Tells the sending system what form our response will be formatted in
    response.setStatus(statusReturned.http_status); // We are figuring this out in the script include
    var writer = response.getStreamWriter();
    writer.writeString(JSON.stringify(statusReturned));

})(request, response);
```

Pull the content out of the received message

Send it to a Script Include to return an error/OK message

Respond to the source system with the custom message

© 2018 ServiceNow All Rights Reserved

REST Endpoint → Script Include → Import Set → Transform

Completed Data Set and Response

Source A		
	employeeNum	
	firstName	
	profile:	
		jobTitle
		businessTitle
	location	
Source B		
	empID	
	f_name	
	locations:[*loop*]	
		primary
		loc

	Import Set Row Field name	Value	Source
User	*uniqueIdentifier*	123	*Source A*
	firstName	Robert	*Source B*
	title	Head Honcho	*Source A*
	loc	US430	*Source A*
Multi Location	*location*	US430	*Source B*
	user	123	*X3 Records*

201 Created sent back. I can decide that a fail on multi-locations is non-critical and include as FYI

User Transform created new Profile

Two many-to-many user locations added, 1 **failed** due to bad location code

© 2018 ServiceNow All Rights Reserved

Other Use Cases @ IHG

Barcode Scanner – Check In via Excel

- Take existing tool sets and processes to easily bring into ServiceNow
- Ensure all of your incoming requests have fully expected data even if you cant control the UI where the request is originating

Change API for tool automation

- Standard Changes and complex workflow triggered through parameters and stages
- Simple and controlled API interface that manages complex
- Manage and customize additional endpoints or parameters that can treat sources differently without rewriting your foundations

© 2018 ServiceNow All Rights Reserved

Other Use Cases @ IHG

Ticket Creation API

- Single enterprise API for platforms to create tickets into ServiceNow
- Ensures adherence to creation and field policies and requirements
- Can create on any Task extended Table

Custom Response


```
{
    "http_status": "201",
    "status_message": "INC0810588 has been created successfully. Use the ticket_number or link for accessing the created record",
    "ticket_number": "INC0810588",
    "sys_id": "8f82970edb2c5b4029c804c2ca961911",
    "link_internal": "https://[url].service-now.com/nav_to.do?uri=incident.do?sys_id=[sysID]",
    "link_customer": "https://[url].service-now.com/cms/tickets_detail.do?sysparm_document_key=incident,[sysID]",
    "table": "incident"
}
```


© 2018 ServiceNow All Rights Reserved

Top Takeaways

1

ADDING SCRIPTED REST ALLOWS YOU TO CODE BEFORE IMPORT SET ROW

2

USE CUSTOM ERROR HANDLING AND DATA SANITATION FOR ENTERPRISE LEVEL APIS

3

IMPORT SET ROWS TO UTILIZE SERVICENOW'S GREAT EXISTING TOOL SET

© 2018 ServiceNow All Rights Reserved

Q&A

Take This Code Home

- https://github.com/ToneyTime/ServiceStartsNow-ScriptedREST
  - Short URL: https://goo.gl/MZt5GN
- ServiceStartsNow.com

© 2018 ServiceNow All Rights Reserved

Thank You

Toney Vecchio

Sr. ServiceNow Engineer

ServiceStartsNow.com

Toney.Vecchio@gmail.com

#Know18

© 2018 ServiceNow All Rights Reserved