knowledge14
April 27–May 1, 2014 • Moscone Center • San Francisco, CA

# Lab Guide

# Integration 102: Discovering the Power of REST

Chris Maloy & Sebastiaan de Mann

Lab instance: https://**<<name>>**.lab.service-now.com

Login / Passwords:

admin / Knowledge14
itil / Knowledge14
employee / Knowledge14

# Lab Goal

Learn how to submit REST based requests using the REST Message feature in ServiceNow.

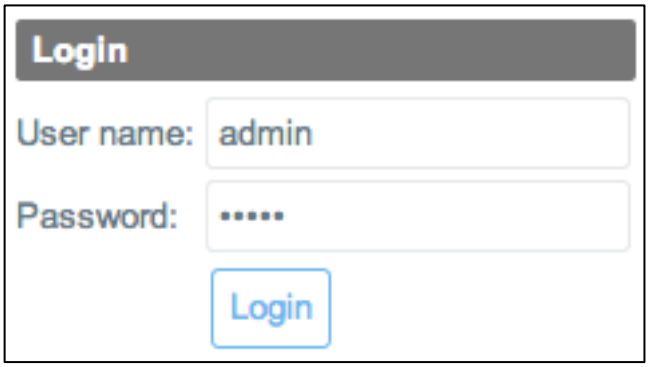
In this exercise, you leverage the GUI REST Message utility to make REST-based requests to a third party ticketing system to create a ticket and retrieve its information.

**Lab 1.1**
**REST**
**Message**

## Log on to Your Training Instance

1. Your instance URL is on the cover page.

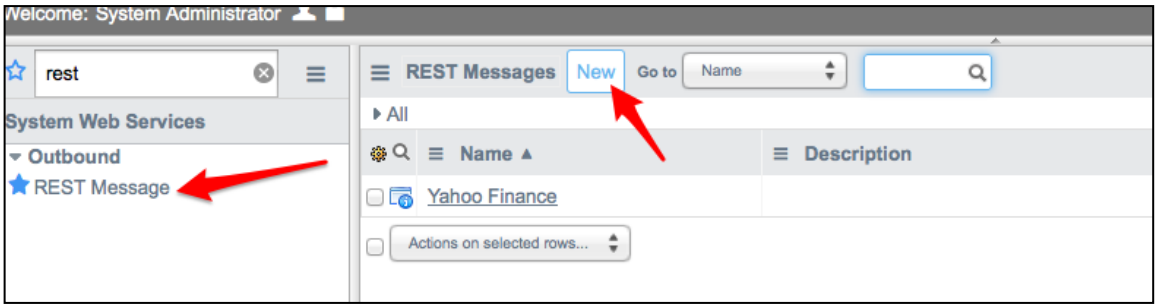
2. Login using the **admin** credentials provided on the cover.




®2014 ServiceNow, Inc. All rights reserved.

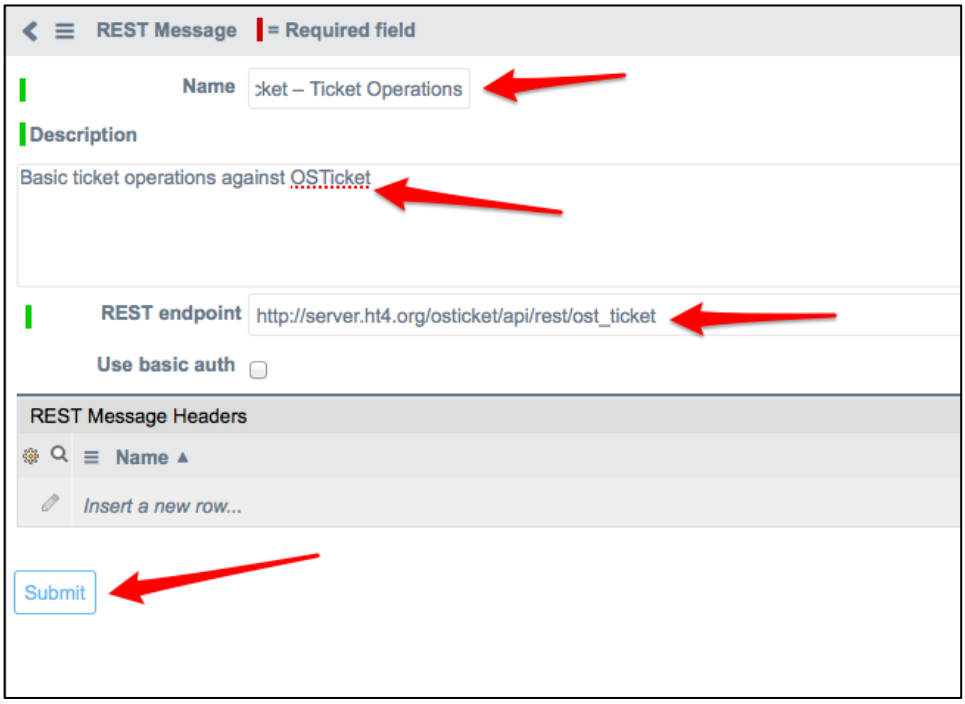
## Configure REST Message

1. In your ServiceNow instance, navigate to **System Web Services > Outbound > REST Message** and click **New**.




3. Fill in the following values for the fields:

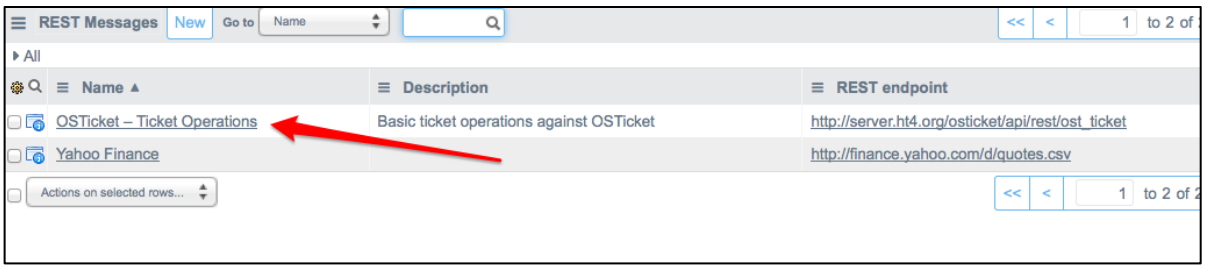
   Name: **OSTicket – Ticket Operations**
   Description: **Basic ticket operations against OSTicket**
   REST endpoint: **http://server.ht4.org/osticket/api/rest/ost_ticket**



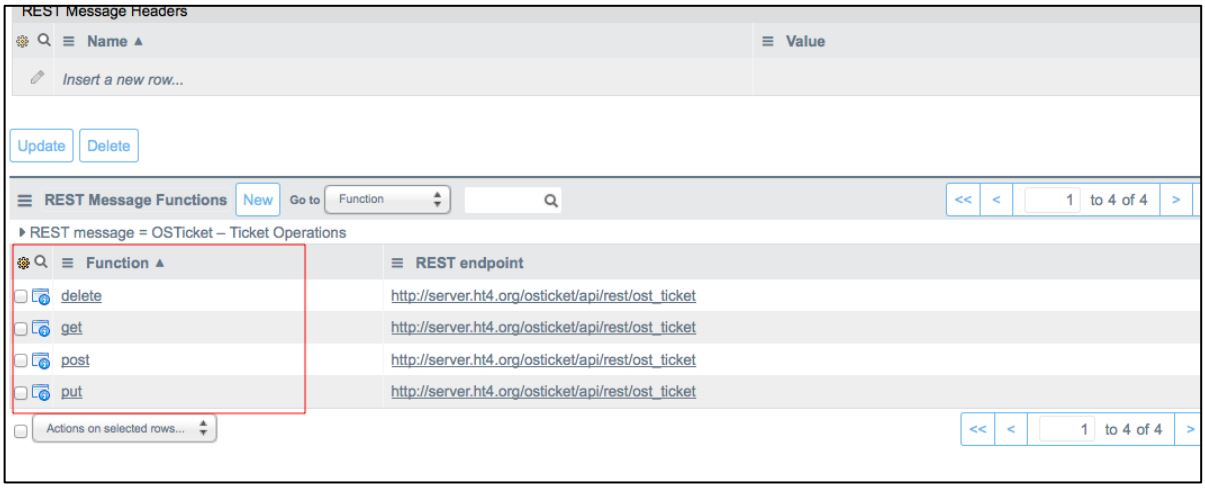

©2014 ServiceNow, Inc. All rights reserved.

4. Click **Submit**.

5. Once the new REST Message is listed, click **OSTicket – Ticket Operations** to open the record.



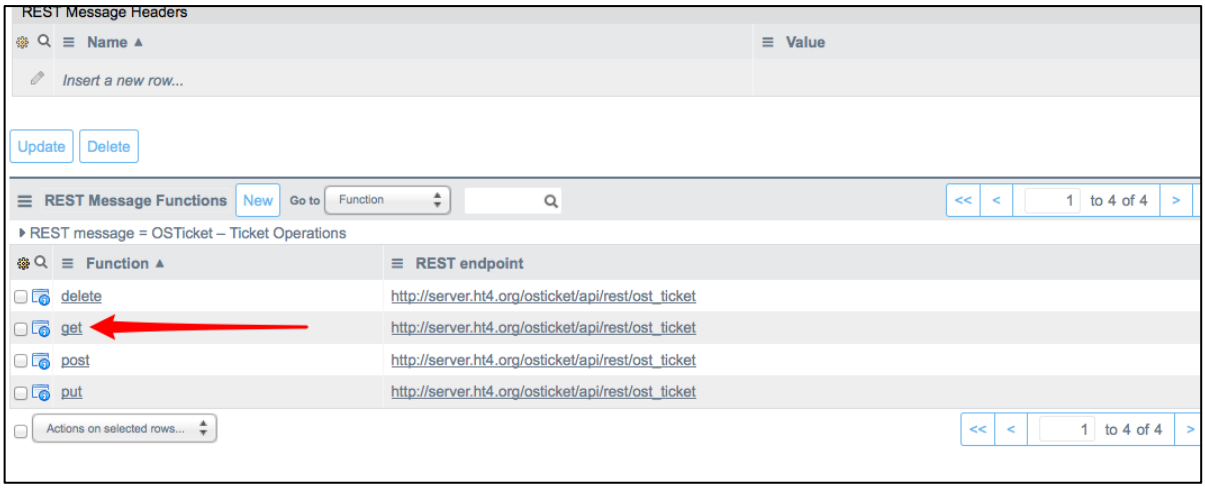

6. Verify that the four basic HTTP operations are listed as **REST Message Functions**.




©2014 ServiceNow, Inc. All rights reserved.

## Set up a basic ticket query

1. Open the **get** REST Message Function inside of the **OSTicket – Ticket Operations** Rest Message record.



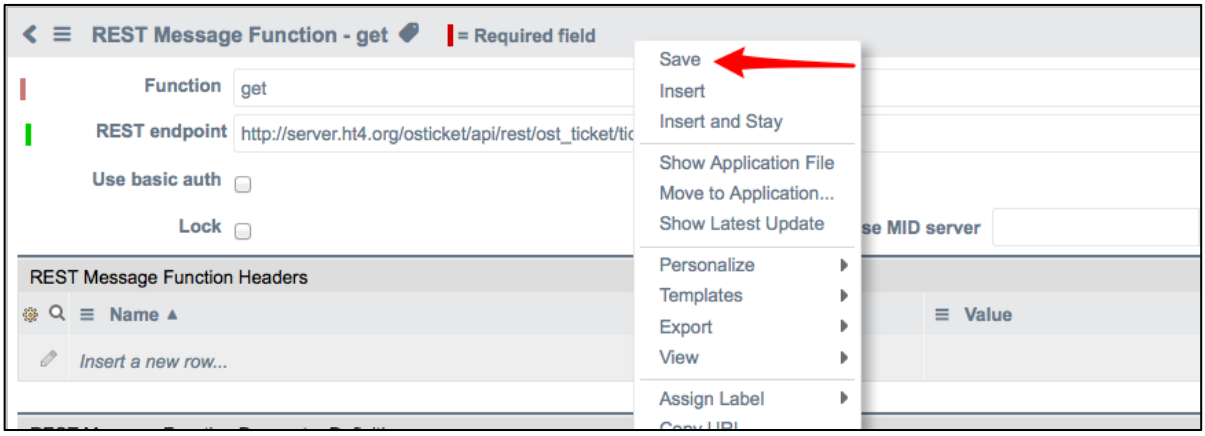

2. The API specification with this web service is to use “ticketID/TICKET_NUM”. Since TICKET_NUM changes depending on which ticket you create, you want to use a variable for it in the endpoint.

3. Modify the **REST Endpoint** to handle a variable-based ticket query by adding the following path to the current REST Endpoint: **/ticketID/${ticketNum}**. To modify the REST endpoint field, you need to click the padlock next to the label.




©2014 ServiceNow, Inc. All rights reserved.

4. Right-click the form header and **Save**.



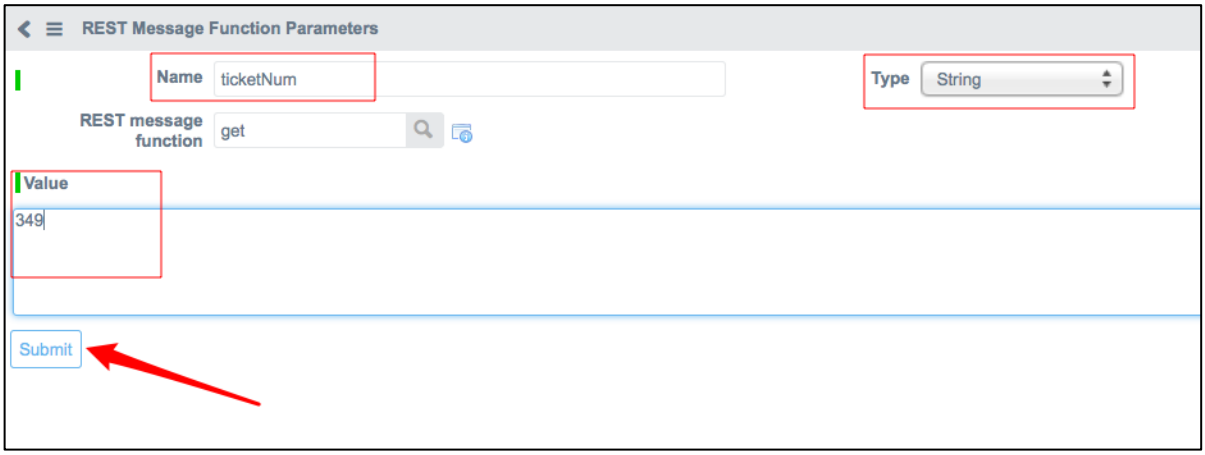

5. Now create a test value for the variable you created in the endpoint to run sample requests. In the **REST Message Function Parameters** section of the form, click **New**.




©2014 ServiceNow, Inc. All rights reserved.

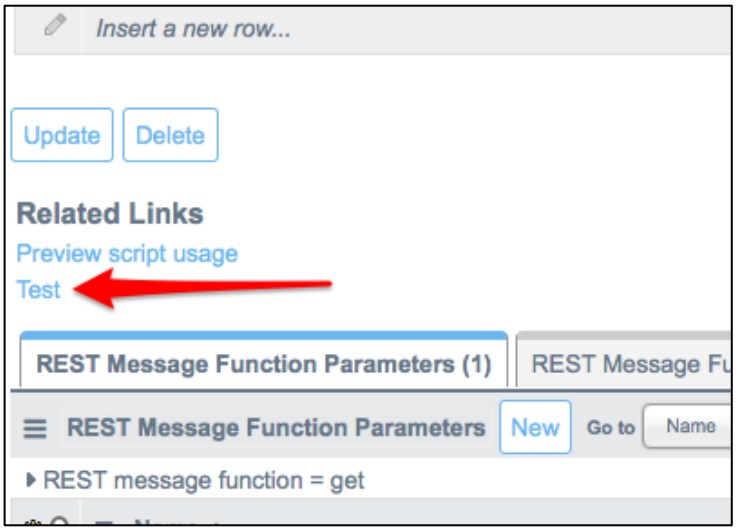
6. Complete the form as follows:

   Name: **ticketNum**
   Type: **String**
   Value: **349**



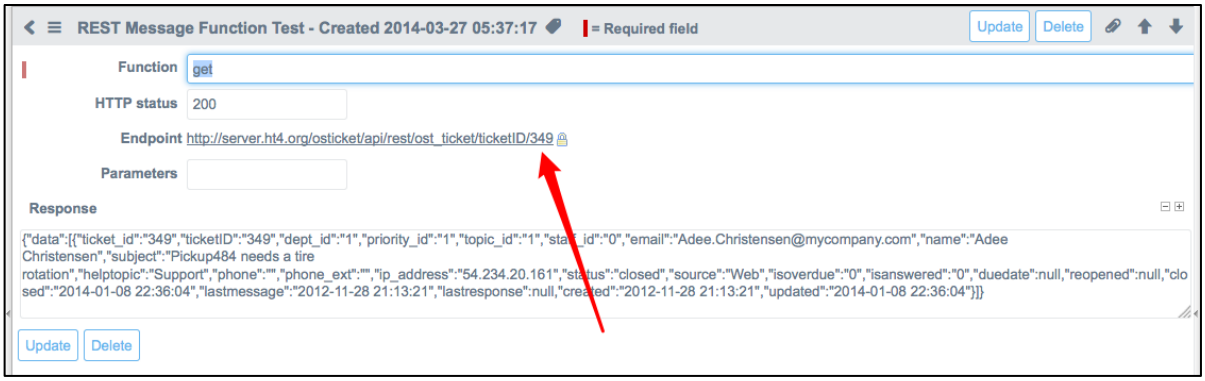

7. Click **Submit**.

8. Now that you have a sample variable, click the **Test** link on the **REST Message Function** form.




©2014 ServiceNow, Inc. All rights reserved.

9. You are redirected to a REST Message Function Test form that contains the Response details. Notice how the variable, **${ticketNum}**, in the Endpoint was replaced by the sample value you specified.



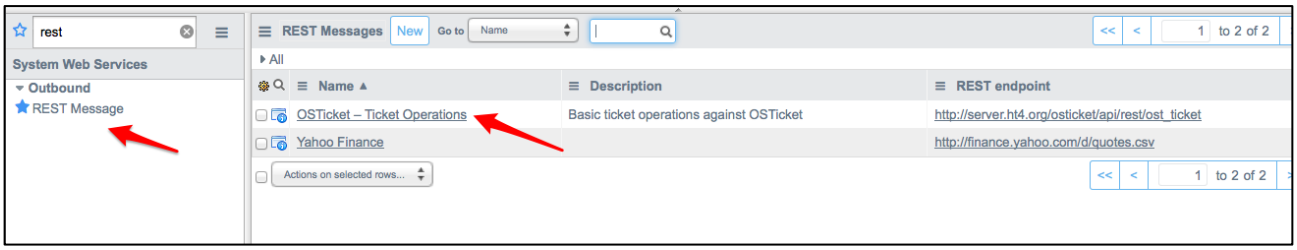

10. The **Response** field contains the JSON string that represents the data within the ticket in the OSTicket system. If you were to format that string, it would look like:


```
{
   "data":[
      {
         "ticket_id":"349",
         "ticketID":"349",
         "dept_id":"1",
         "priority_id":"1",
         "topic_id":"1",
         "staff_id":"0",
         "email":"Adee.Christensen@mycompany.com",
         "name":"Adee Christensen",
         "subject":"Pickup484 needs a tire rotation",
         "helptopic":"Support",
         "phone":"",
         "phone_ext":"",
         "ip_address":"54.234.20.161",
         "status":"open",
         "source":"Web",
         "isoverdue":"0",
         "isanswered":"0",
         "duedate":null,
         "reopened":null,
         "closed":null,
         "lastmessage":"2012-11-28 21:13:21",
         "lastresponse":null,
         "created":"2012-11-28 21:13:21",
         "updated":"0000-00-00 00:00:00"
      }
   ]
}
```

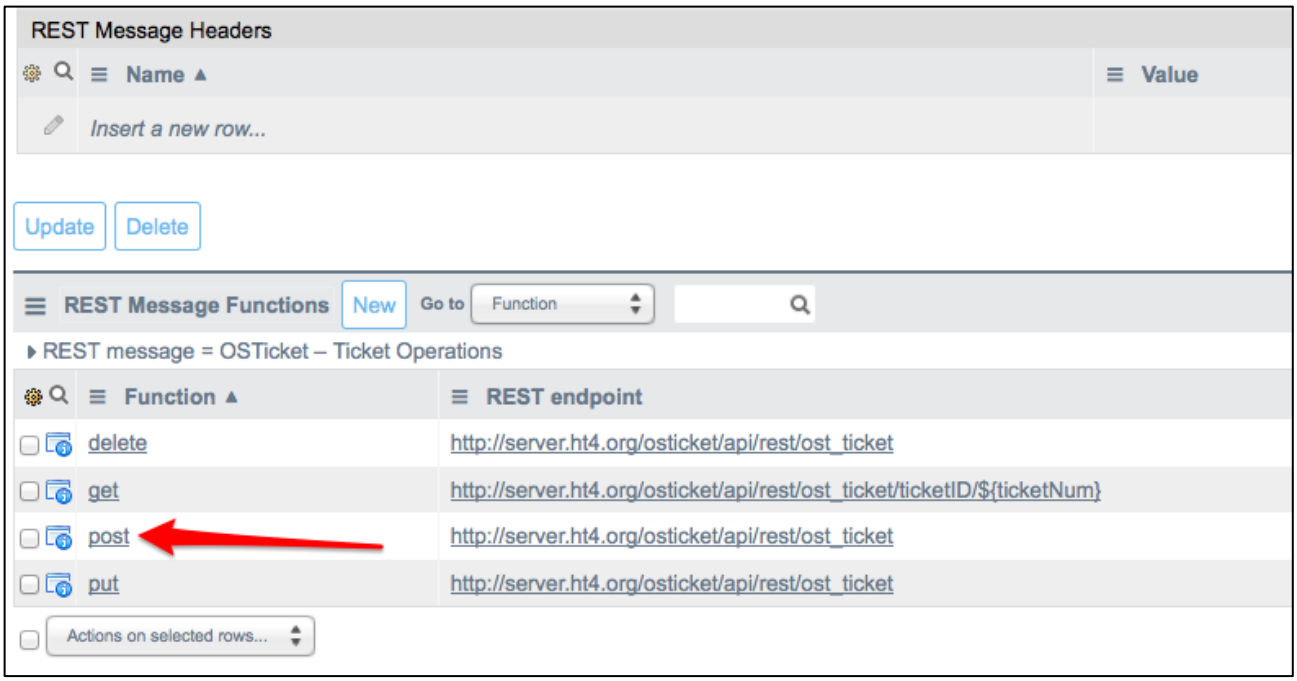

©2014 ServiceNow, Inc. All rights reserved.

## Set up a REST Request to create a ticket

1. Navigate to **System Web Services > Outbound > REST Message** and open the **OSTicket – Ticket Operations** record.




2. The OSTicket REST API requires an HTTP POST method to be used to create tickets. To set this up, click on the **post** REST Message Function.



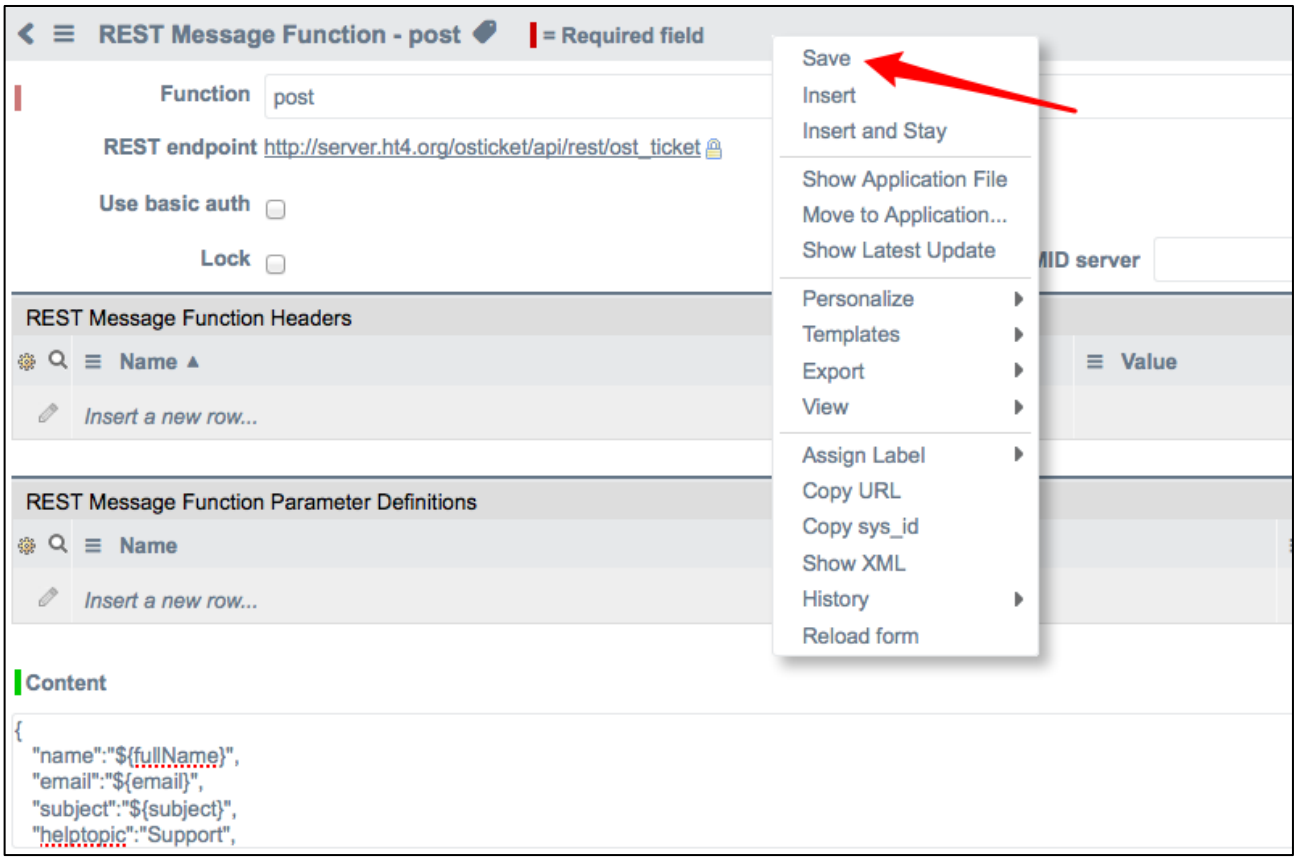

©2014 ServiceNow, Inc. All rights reserved.

3. The OSTicket API requires a JSON formatted string in the post content that contains field / value pairs. To do this, paste the following string in the **content** form of the REST Message Function.

```
{
"name":"${fullName}",
"email":"${email}",
"subject":"${subject}",
"helptopic":"Support",
"ticketID":"${ticketNum}"
}
```

4. Notice that you have inserted some variable strings in the content body so that you can insert data of your choice during each transaction.

5. Right-click the form header and **Save**.



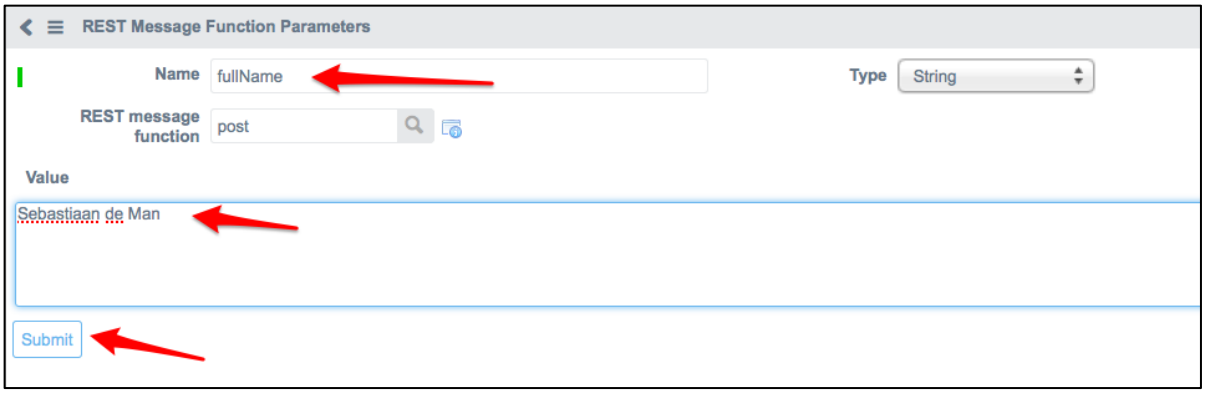

©2014 ServiceNow, Inc. All rights reserved.

6. Now create test variables for each variable in the JSON string. In the **REST Message Function Parameters** related list, click **New**.




7. Complete the form as follows:

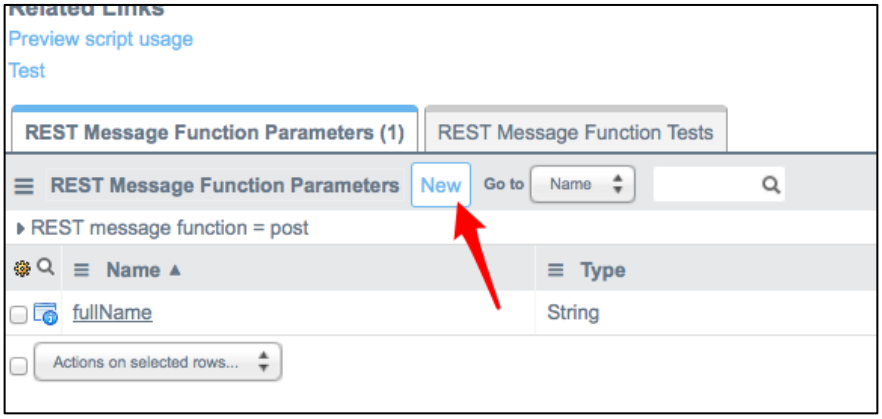
   Name: **fullName**
   Value: ***<your full name>***




8. Click **Submit**.

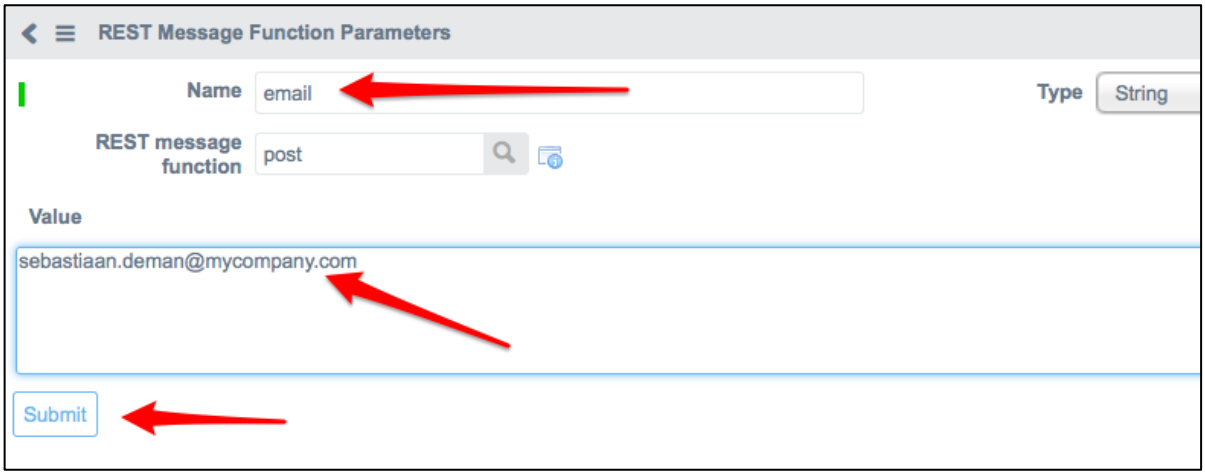
©2014 ServiceNow, Inc. All rights reserved.

9. In the **REST Message Function Parameters** related list, click **New**.




10. Complete the form as follows:

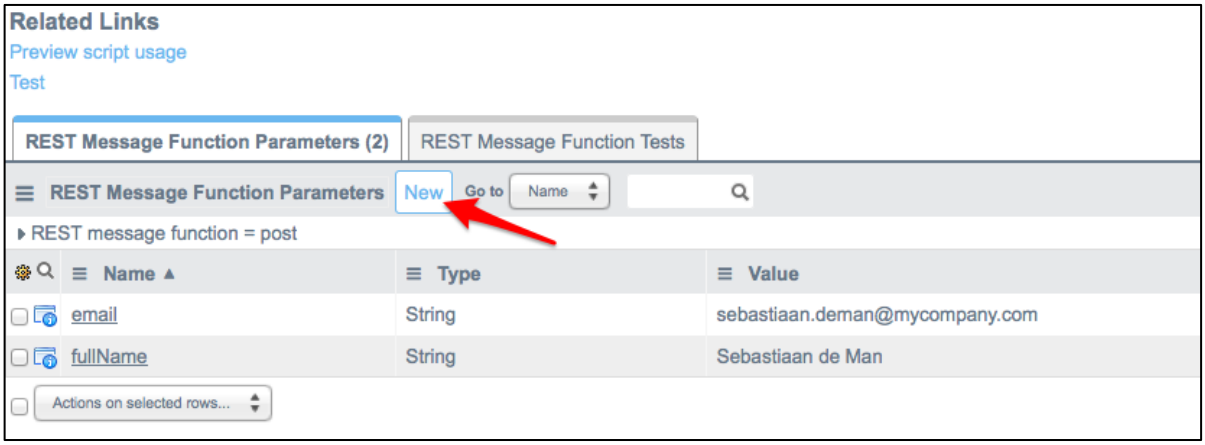
    Name: **email**
    Value: ***<your email address>***




11. Click **Submit**.

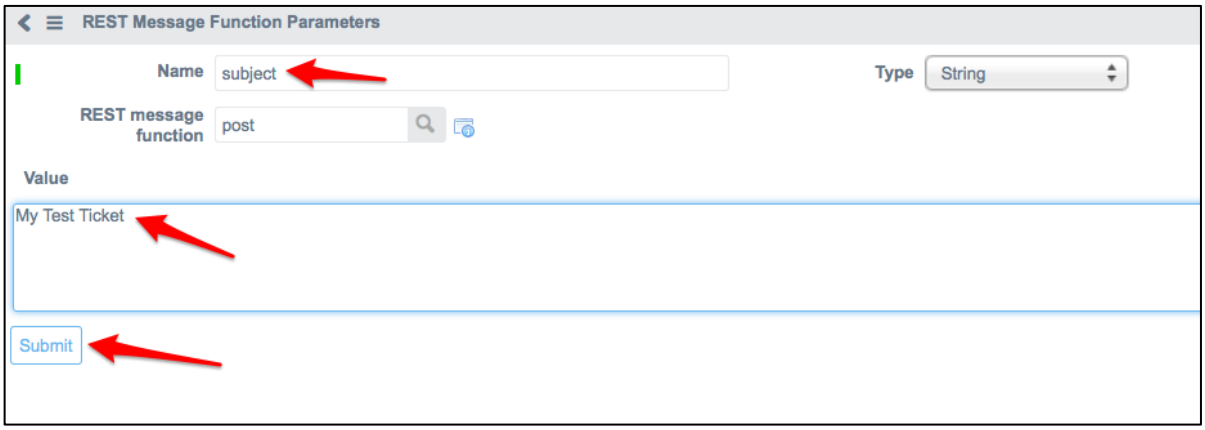
®2014 ServiceNow, Inc. All rights reserved.

12. In the **REST Message Function Parameters** related list, click **New**.




13. Complete the form as follows:

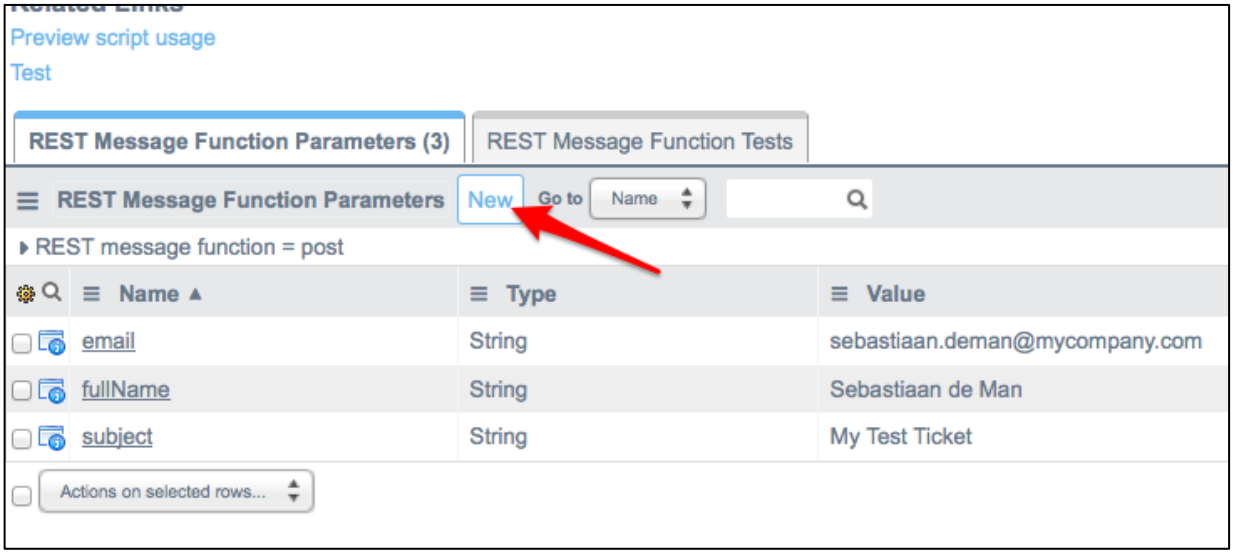
    Name: **subject**
    Value: **My Test Ticket**




14. Click **Submit**.

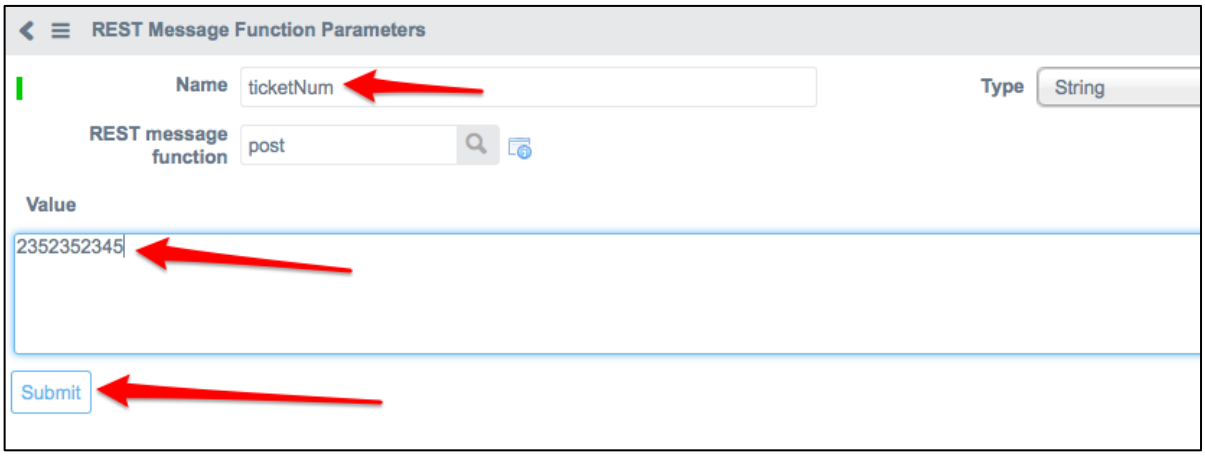
©2014 ServiceNow, Inc. All rights reserved.

15. In the **REST Message Function Parameters** related list, click **New**.




16. Complete the form as follows:

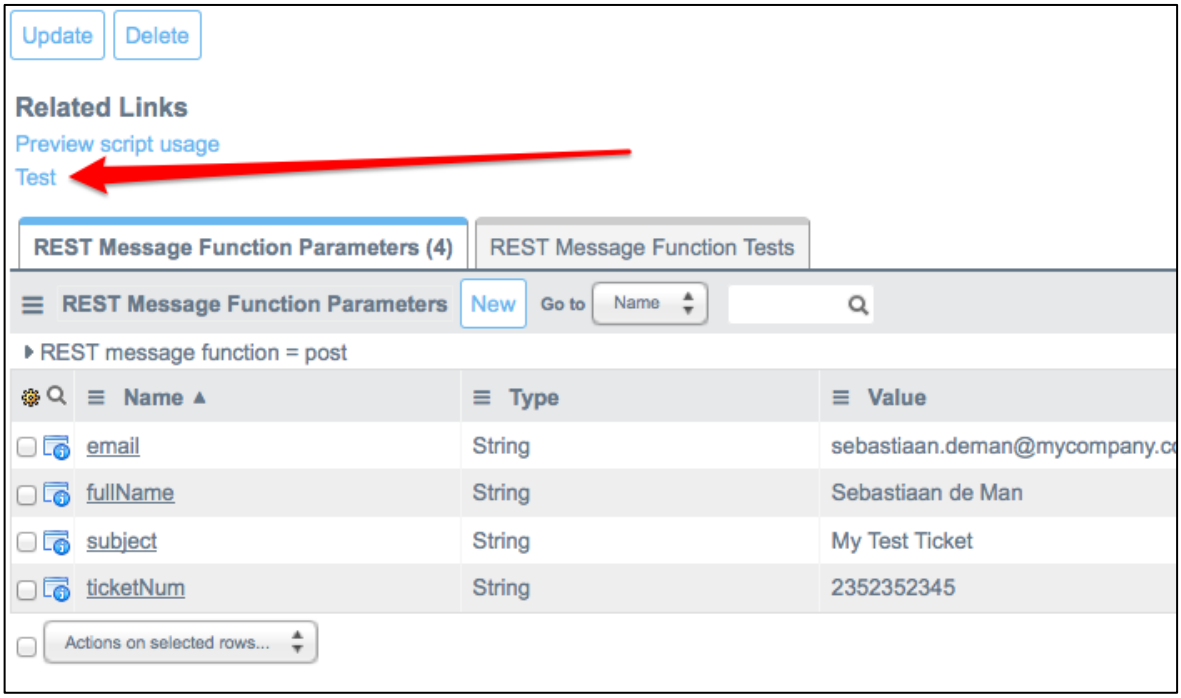
    Name: **ticketNum**
    Value: ***<some random number>*** (make note of this for later verification)




17. Click **Submit**.

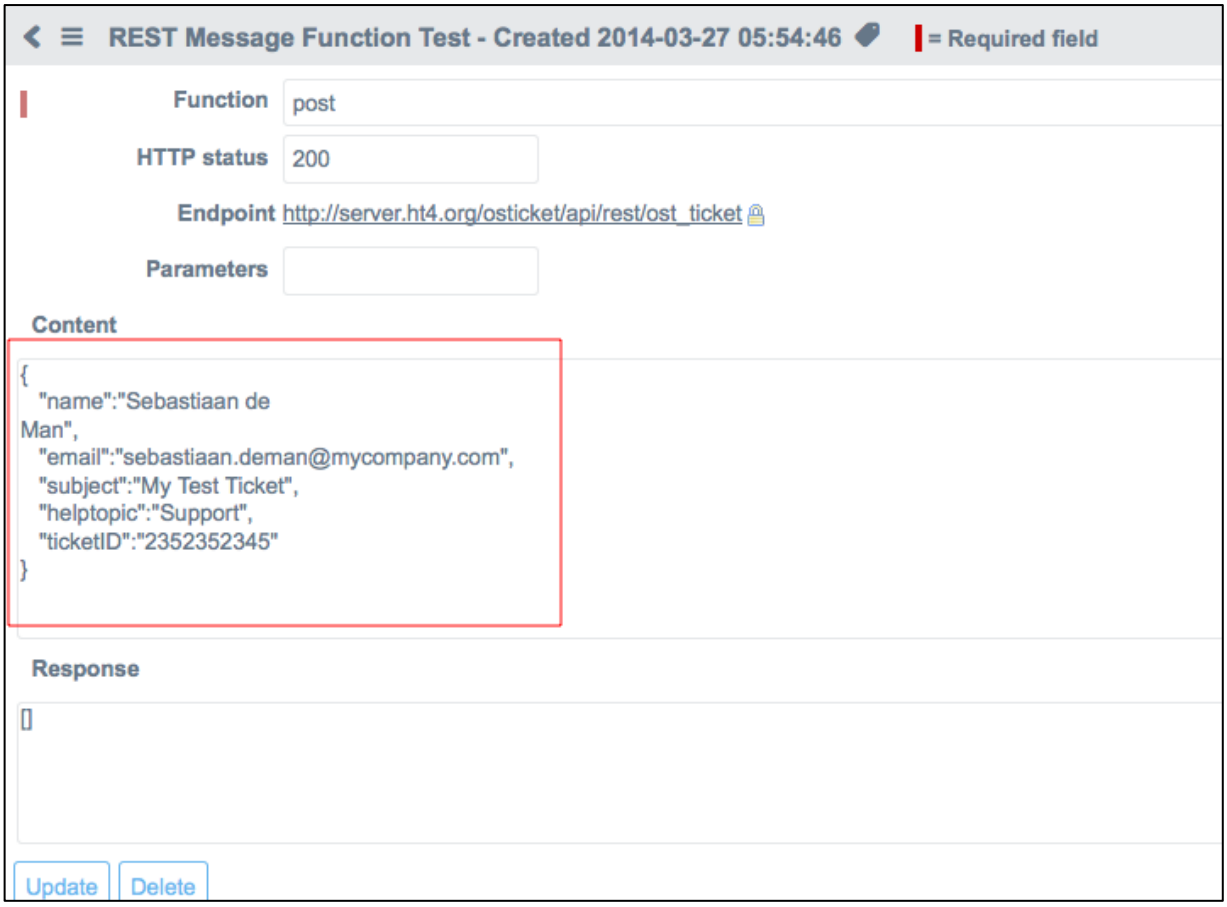
®2014 ServiceNow, Inc. All rights reserved.

18. Under Related Links, click **Test** to submit the information.




®2014 ServiceNow, Inc. All rights reserved.

19. When the response page loads, you will notice that the **Content** field contains the actual body content that was posted to the server. The variables have been replaced with the values set in the previous few steps.

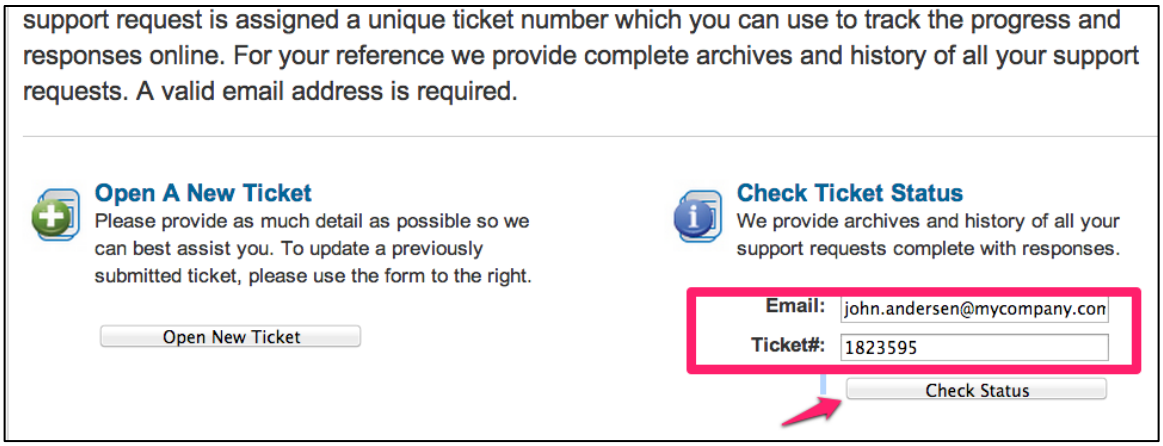



20. Unfortunately, this API does not return a descriptive response. However, you can check the OSTicket user interface to verify that the ticket was created.

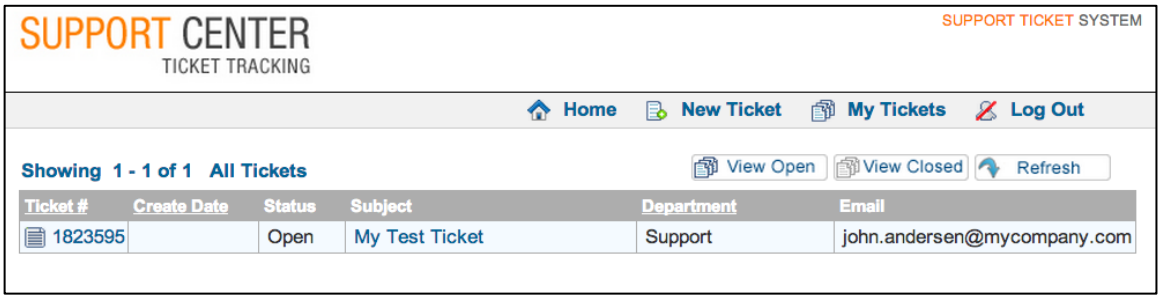
21. Open up a new browser tab or window and go to the following URL: **http://server.ht4.org/osticket/**

©2014 ServiceNow, Inc. All rights reserved.

22. In the login box, enter the email address you used in the REST request as well as the ticketNum variable you used.




23. Your ticket should be listed in the system.




24. Click **Log Out**.

©2014 ServiceNow, Inc. All rights reserved.

# Lab Goal

Learn how to leverage REST Message calls within ServiceNow scripts to provide dynamic communication between systems.

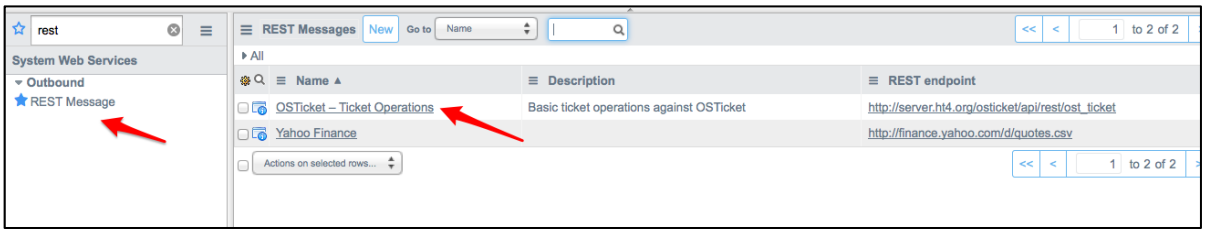
In this exercise, you leverage REST Message functions from within a business rule to create tickets in a third party ticketing system.

**Lab 2.1 Scripted REST Message**

## Generate a sample REST Message Function script

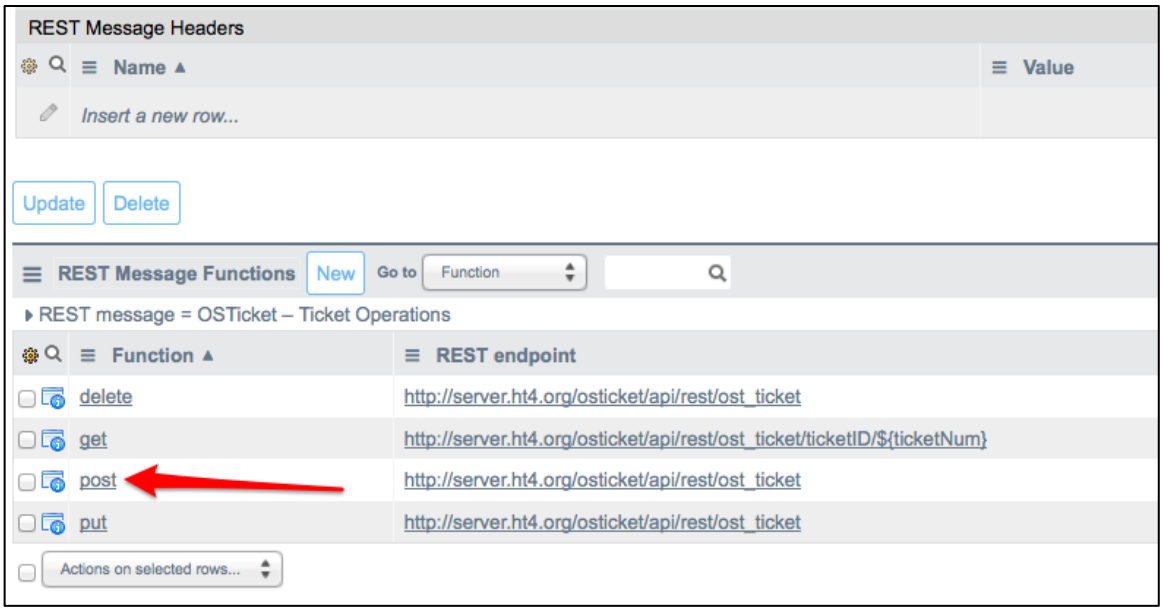
1. The REST Message Function tool auto generates a sample REST Message script based on function variables defined in the REST Message Function Parameters list on the REST Message Function form.

2. Navigate to **System Web Services > Outbound > REST Message** and open the **OSTicket – Ticket Operations** record.

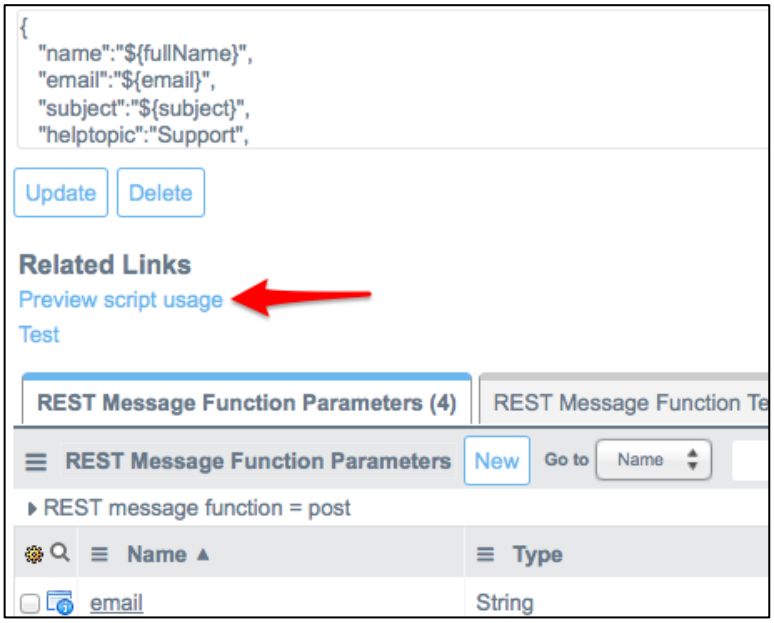



©2014 ServiceNow, Inc. All rights reserved.

3. In this exercise, you create a ticket in the third party system. You need to leverage the **post** function to do this. In the **Rest Message Functions** related list, click **post**.

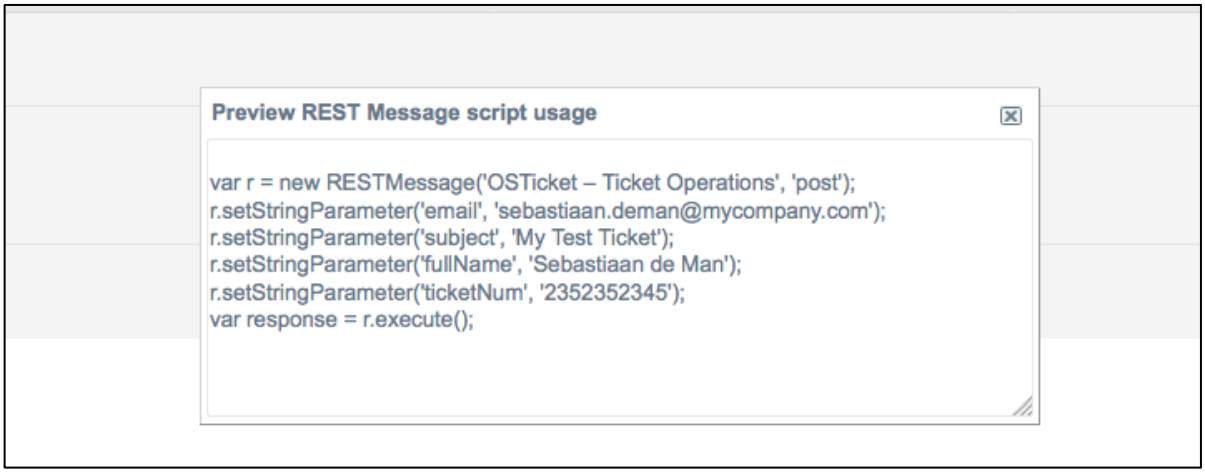



4. Since you have already defined sample REST Message Function Parameters, all you need to do is click the **Preview script usage** Related Link on the form.




©2014 ServiceNow, Inc. All rights reserved.

5. Copy the script to your clipboard for future use.

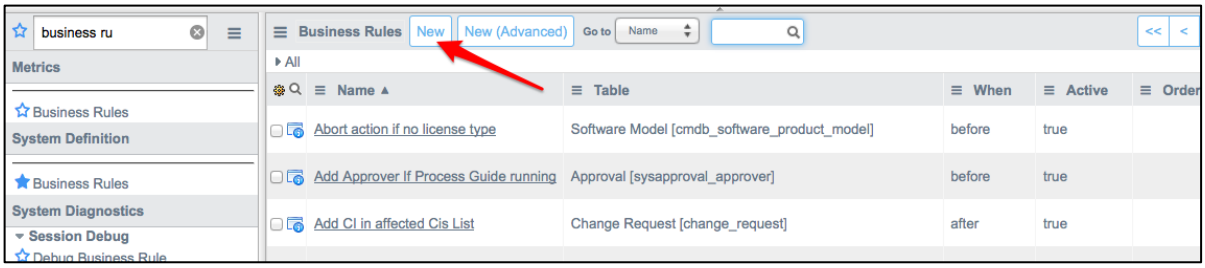



6. Close the Preview REST Message script usage dialog box.

## Create a Business Rule on the Incident Table

When a new incident is created, you want to create a corresponding incident in the OSTicket system.

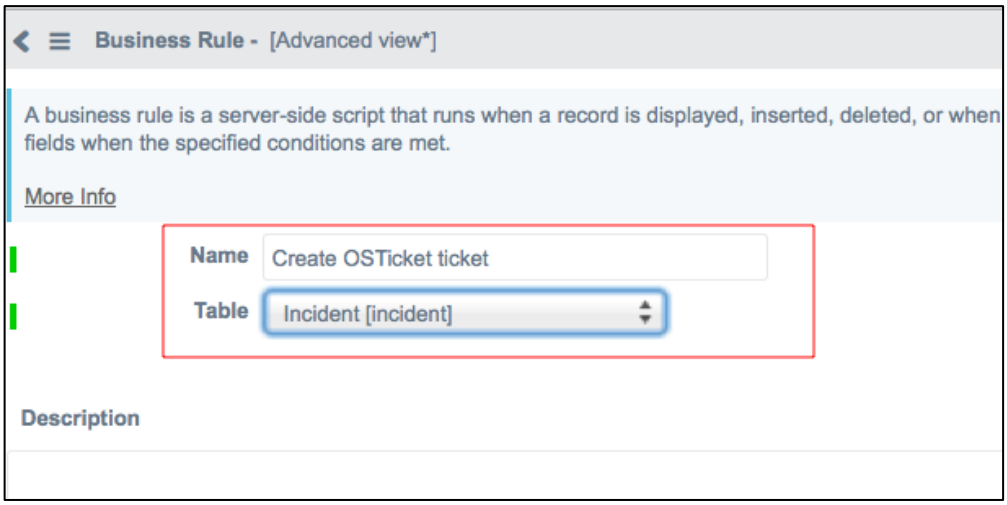
1. Navigate to **System Definition > Business Rules** and click **New**.




©2014 ServiceNow, Inc. All rights reserved.

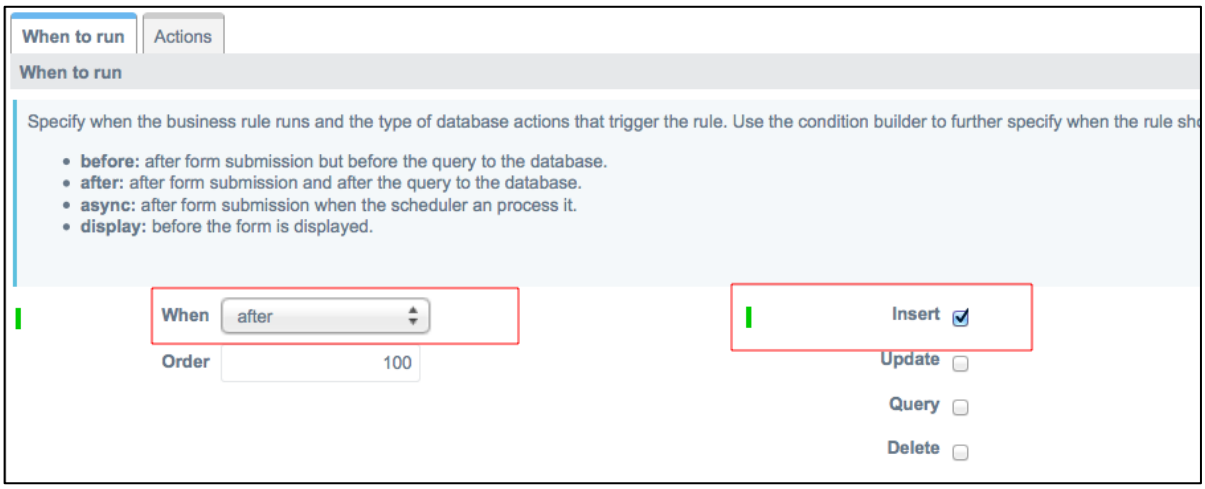
2. Complete the fields with the following information:

   Name: **Create OSTicket ticket**
   Table: **Incident [incident]**



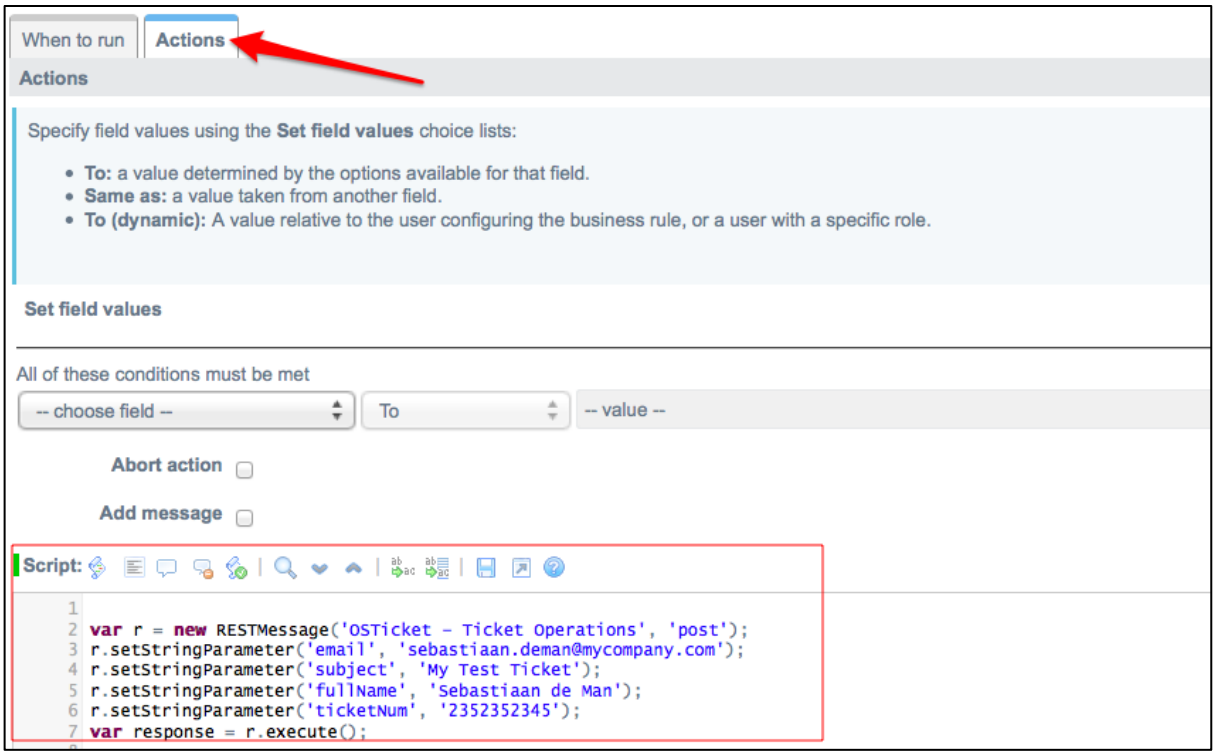

3. In the **When to run** form section, complete the fields with the following information:

   When: **after**
   Insert**: Checked**




©2014 ServiceNow, Inc. All rights reserved.

4. In the **Actions** form section, paste the auto-generated code from the REST Message Function into the Script field.




®2014 ServiceNow, Inc. All rights reserved.

5. Modify the script so that it uses **Incident** values rather than the hard coded strings you used in testing the function. Use the following Incident fields to set the REST variables in the script:

   subject: **current.short_description**
   fullName: **current.caller_id**
   ticketNum: **current.number**

```
1
2 var r = new RESTMessage('OSTicket - Ticket Operations', 'post');
3 r.setStringParameter('email', 'sebastiaan.deman@mycompany.com');
4 r.setStringParameter('subject', current.short_description);
5 r.setStringParameter('fullName', current.caller_id);
6 r.setStringParameter('ticketNum', current.number);
7 var response = r.execute();
8
9
```

   NOTE: the email address is still hardcoded. This allows you to log into the OSTicket system later with your original email address to verify ticket creation in a later step.

6. Click **Submit** to save the business rule.

## Test this portion of the integration

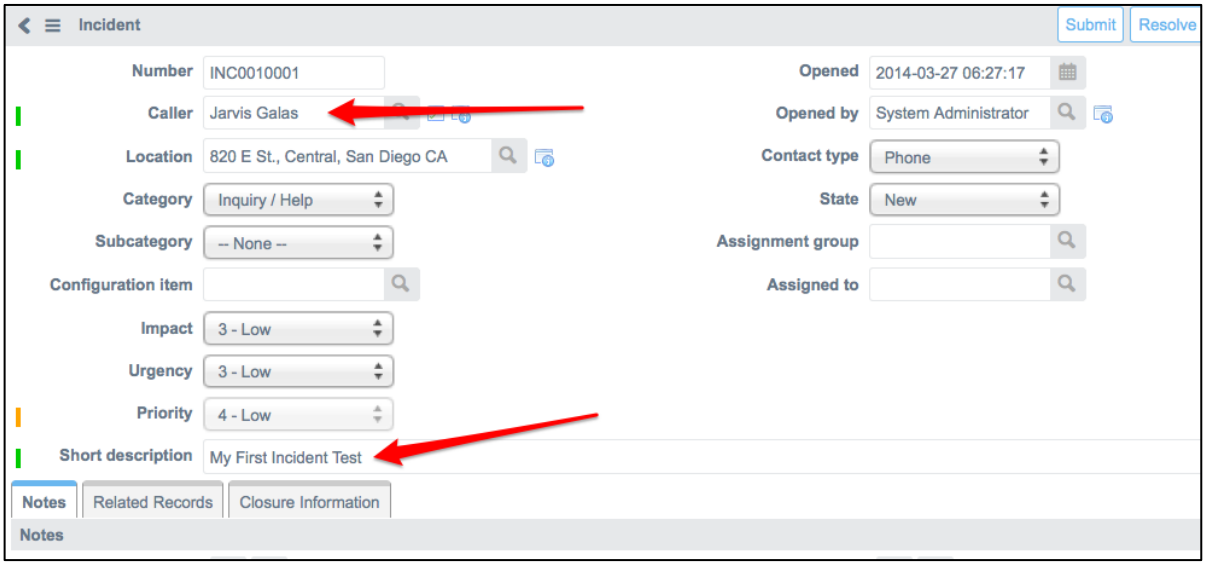
Create an incident to verify that whatever incident is created in ServiceNow also gets created in the OSTicket system.

1. Navigate to **Incident > Create New**.

2. Set the Caller to **Jarvis Galas**.

©2014 ServiceNow, Inc. All rights reserved.

3. Set the Short Description to be **My First Incident Test**.



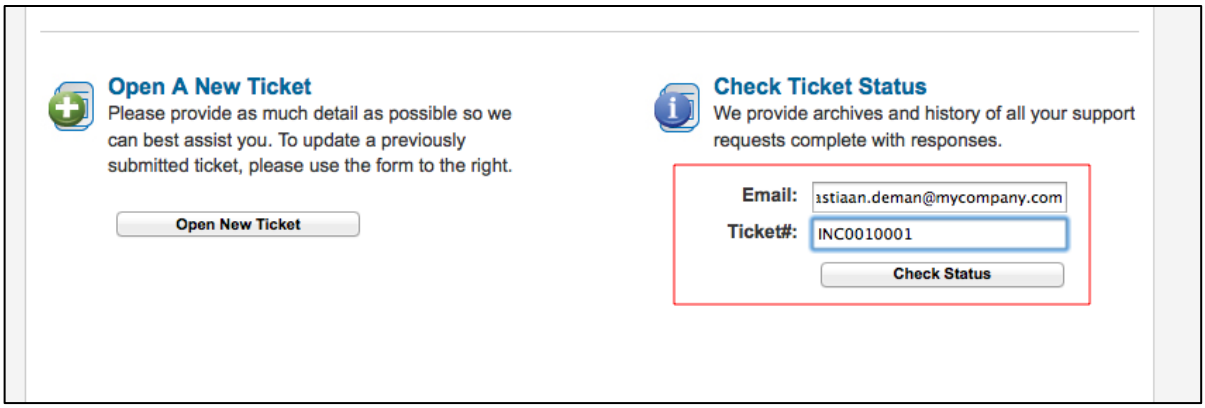

4. Copy the Incident **Number** to the clipboard for a future step.

5. Click **Submit**.

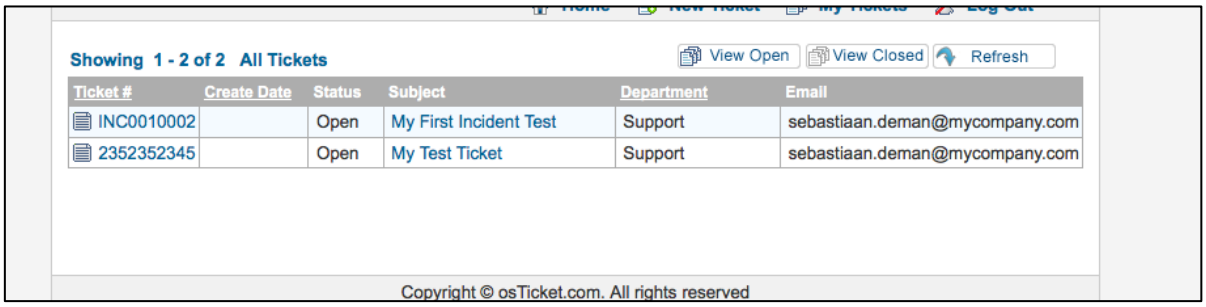
6. Open up a new browser tab or window and go to the following URL:
   **http://server.ht4.org/osticket/**

©2014 ServiceNow, Inc. All rights reserved.

7. In the login box, enter the email address you used the business rule and the Incident Number that you copied from the ServiceNow incident record and click **Check Status**.




8. Your ticket should be listed in the system.



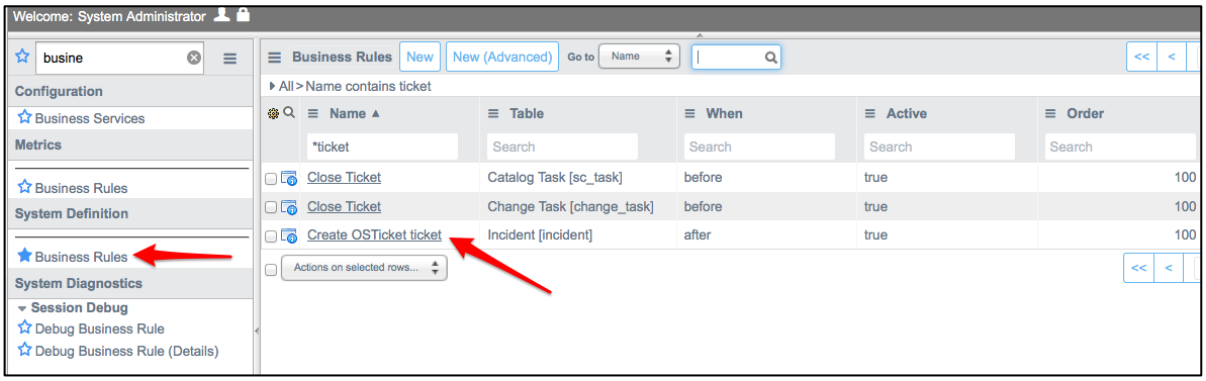

9. Click **Log Out**.

©2014 ServiceNow, Inc. All rights reserved.

## Set up the business rule to store the OSTicket GUID for the ticket

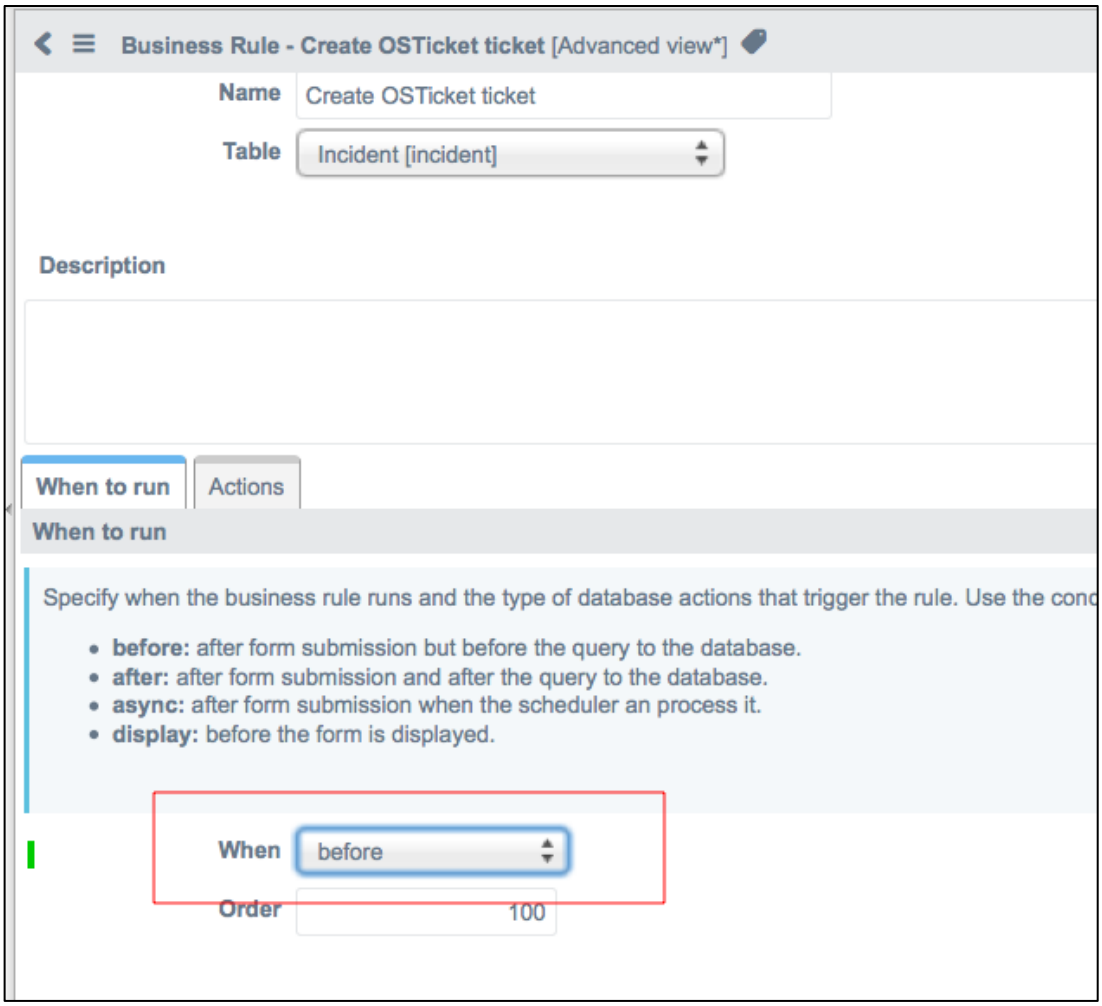
You now want to store the internal Globally Unique Identifier (GUID) number of the OSTicket that was created in the third party system should you need to communicate with OSTicket in the future. OSTicket uses a **ticket_id** field for its unique GUID. The similar **ticketID** field in the OSTicket system is the display value of the ticket number.

1. Navigate to **System Definition > Business Rules** and open the **Create OSTicket ticket** record.




©2014 ServiceNow, Inc. All rights reserved.

2. You want to set data on to the incident in ServiceNow, so instead of having the Business Rule run **after** the data is saved, you want to set the **When** choice value to be **before**.




3. Append code to the **Script** field that calls the **get** REST Message Function to get details on the incident that was just created.

```
//Make a query request for the ticket that was just created
var details = new RESTMessage('OSTicket - Ticket Operations','get');
details.setStringParameter('ticketNum', current.number);
var detailsResponse = details.execute();
```

©2014 ServiceNow, Inc. All rights reserved.

4. You need to then get the response body string from the response object.

```
//Get the response string
var jsonString = detailsResponse.getBody();
```

5. You may recall the response from the OSTicket system will look like the following JSON String:


```
{
   "data":[
      {
         "ticket_id":"349",
         "ticketID":"944852",
         "dept_id":"1",
         "priority_id":"1",
         "topic_id":"1",
         "staff_id":"0",
         "email":"Adee.Christensen@mycompany.com",
         "name":"Adee Christensen",
         "subject":"Pickup484 needs a tire rotation",
         "helptopic":"Support",
         "phone":"",
         "phone_ext":"",
         "ip_address":"54.234.20.161",
         "status":"open",
         "source":"Web",
         "isoverdue":"0",
         "isanswered":"0",
         "duedate":null,
         "reopened":null,
         "closed":null,
         "lastmessage":"2012-11-28 21:13:21",
         "lastresponse":null,
         "created":"2012-11-28 21:13:21",
         "updated":"0000-00-00 00:00:00"
      }
   ]
}
```


©2014 ServiceNow, Inc. All rights reserved.

6. You need to convert this string into a JavaScript object with the following code appended in the **Script** field.

```
//Convert the JSON String to an object

var parser = new JSONParser();
var parsed = parser.parse(jsonString);
```

7. The resulting object will have the following structure:

- parsed (object)
  - data (array)
    - name (string)
    - dept_id (string)
    - ticket_id (string)
    - ....etc....

8. When performing an integration, it is a common practice to set the 3rd party's record GUID to the ServiceNow record as well as a tag that indicates what system the record is integrated with. There are two fields on task tables for this purpose.

   **correlation_id:** this field typically holds the third party GUID of the associated record

   **correlation_display**: this field is usually a hard coded string that indicates what third party system is integrated with this task

9. Append the following code to the **Script** to set the ticket_id GUID to the incident's **correlation_id** field as well as to set the **correlation_display** field to be **OSTicket:**

```
current.correlation_id = parsed.data[0].ticket_id;
current.correlation_lay = "OSTicket";
```

©2014 ServiceNow, Inc. All rights reserved.

10. Your business rule **Script** should now look similar to, but not exactly like, the following (email address should be different):

```
var r = new RESTMessage('OSTicket - Ticket Operations', 'post');
r.setStringParameter('fullName', current.caller_id);
r.setStringParameter('email', 'sebastiaan.deman@mycompany.com');
r.setStringParameter('subject', current.short_description);
r.setStringParameter('ticketNum', current.number);
var response = r.execute();

//Make a query request for the ticket that was just created
var details = new RESTMessage('OSTicket - Ticket Operations', 'get');
details.setStringParameter('ticketNum', current.number);
var detailsResponse = details.execute();

//Get the response string
var jsonString = detailsResponse.getBody();

//Convert the JSON String to an object
var parser = new JSONParser();
var parsed = parser.parse(jsonString);

current.correlation_id = parsed.data[0].ticket_id;
current.correlation_display = "OSTicket";
```

11. Click **Update**.

®2014 ServiceNow, Inc. All rights reserved.

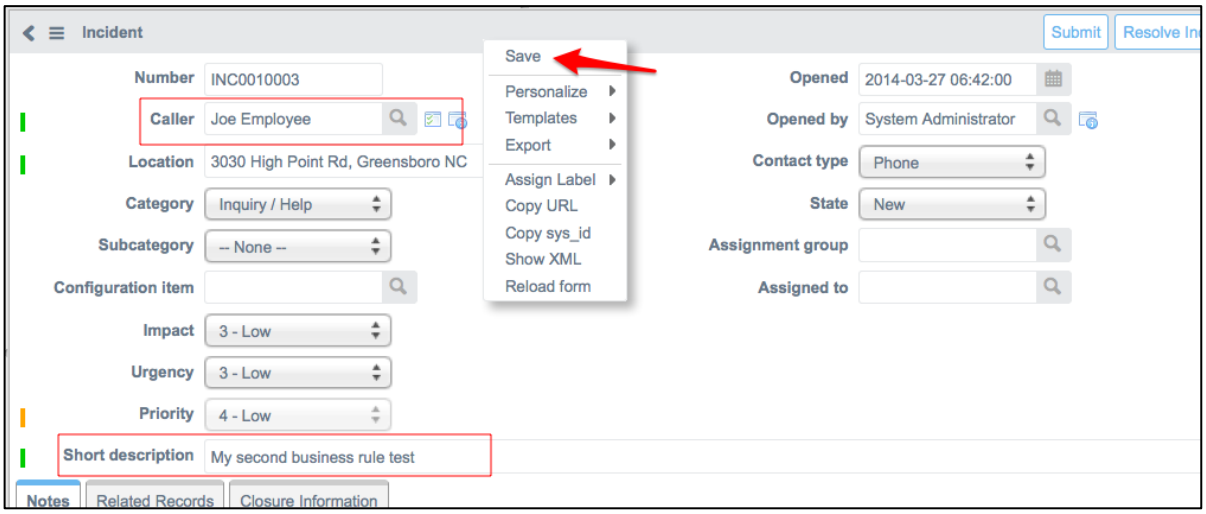
## Test the Integration

1. Navigate to **Incident > Create New**.

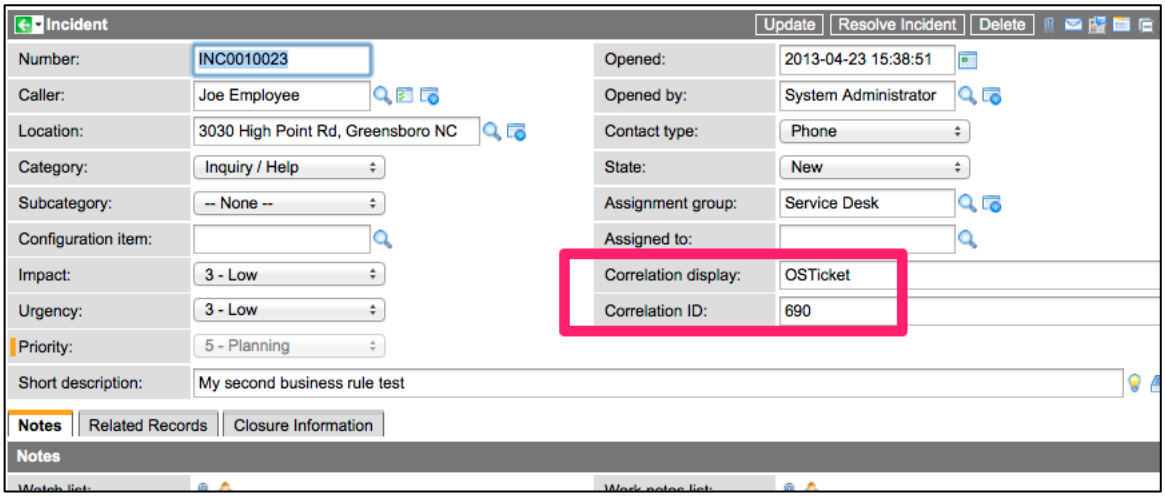
2. Set Caller to **Joe Employee**.

3. Set Short Description to **My second business rule test**.

4. Right-click the form header and **Save**.




5. After the form refreshes, notice the values of the **Correlation display** and **Correlation ID** fields. They should look similar to the screen shot below, but have a different ID:




®2014 ServiceNow, Inc. All rights reserved.

# Lab Goal

Learn how to make RESTful calls into a ServiceNow instance to perform basic query and write operations.

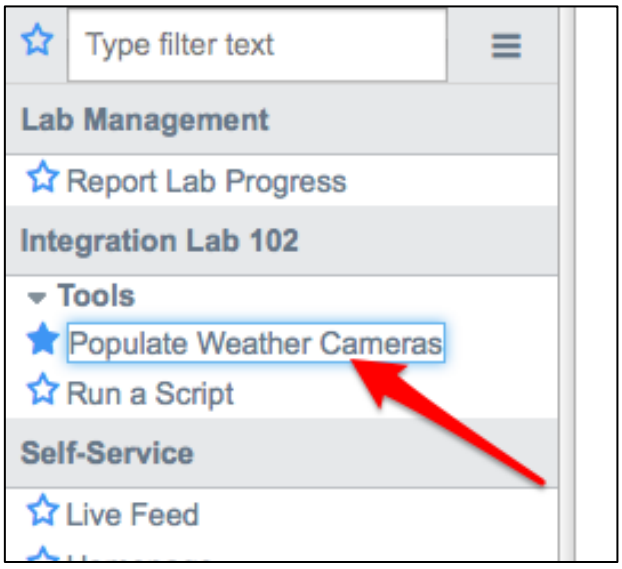
In this exercise, you leverage the JSON Web Service by a third party REST client. From that client you will submit REST calls that query and manipulate a table within ServiceNow.

**Lab 3.1**
**JSON**
**Inbound**
**REST**
**Message**

## Create a Weather Camera record using REST

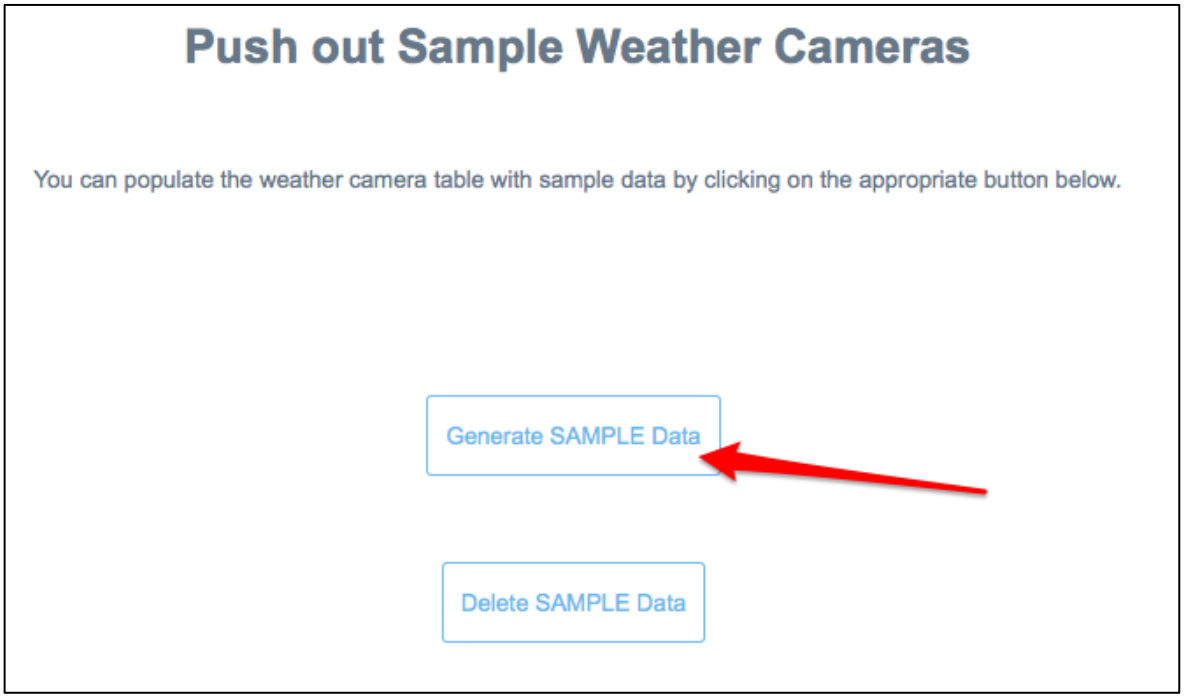
Your demo instance has a custom table called **Weather Cameras** that allows the instance to store records about weather cameras managed by your company. You want to use the REST interface to manipulate the data in that table.

1. First, you need to populate the Weather Cameras table with some demo data. To do this, navigate your ServiceNow instance to **integration Lab 102 > Populate Weather Cameras**.

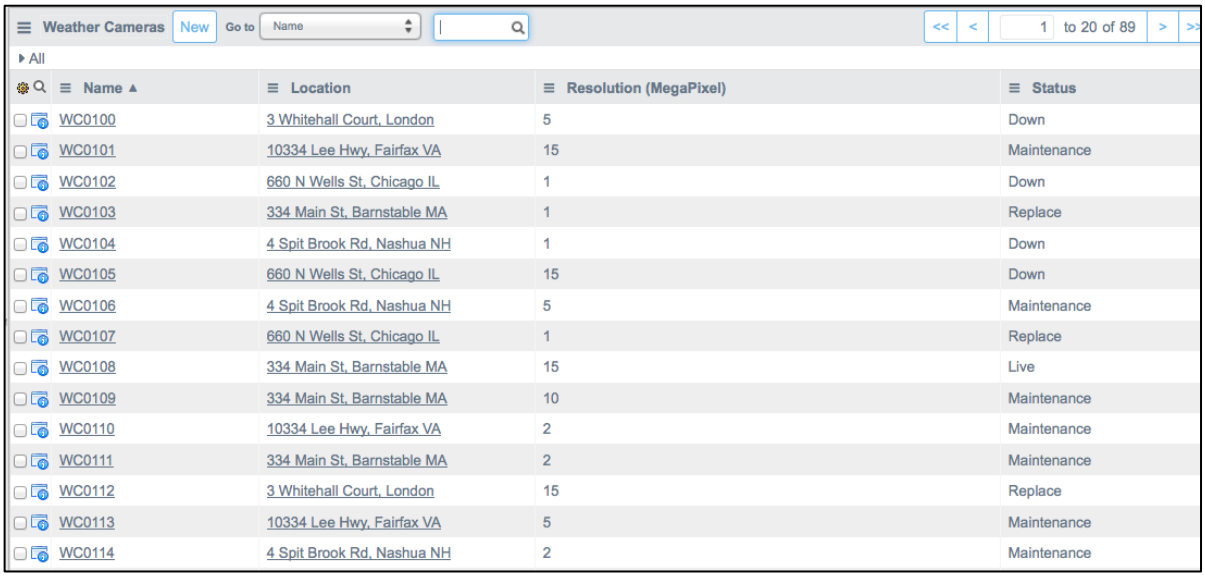



©2014 ServiceNow, Inc. All rights reserved.

2. Click **Generate SAMPLE Data** (Click only once).




3. You are directed to the list of Weather Cameras that were generated.



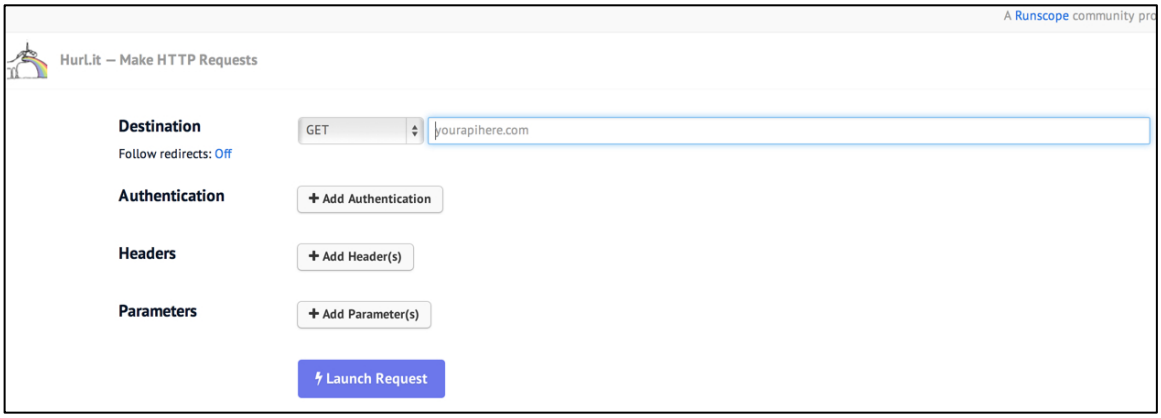

| Name | Location | Resolution (MegaPixel) | Status |
|---|---|---|---|
| WC0100 | 3 Whitehall Court, London | 5 | Down |
| WC0101 | 10334 Lee Hwy, Fairfax VA | 15 | Maintenance |
| WC0102 | 660 N Wells St, Chicago IL | 1 | Down |
| WC0103 | 334 Main St, Barnstable MA | 1 | Replace |
| WC0104 | 4 Spit Brook Rd, Nashua NH | 1 | Down |
| WC0105 | 660 N Wells St, Chicago IL | 15 | Down |
| WC0106 | 4 Spit Brook Rd, Nashua NH | 5 | Maintenance |
| WC0107 | 660 N Wells St, Chicago IL | 1 | Replace |
| WC0108 | 334 Main St, Barnstable MA | 15 | Live |
| WC0109 | 334 Main St, Barnstable MA | 10 | Maintenance |
| WC0110 | 10334 Lee Hwy, Fairfax VA | 2 | Maintenance |
| WC0111 | 334 Main St, Barnstable MA | 2 | Maintenance |
| WC0112 | 3 Whitehall Court, London | 15 | Replace |
| WC0113 | 10334 Lee Hwy, Fairfax VA | 5 | Maintenance |
| WC0114 | 4 Spit Brook Rd, Nashua NH | 2 | Maintenance |

©2014 ServiceNow, Inc. All rights reserved.

4. In this exercise you use a third party REST client create a new Weather Camera record that has the following information:

   Name: **WC0800**
   Resolution: **8**
   Status: **Live**

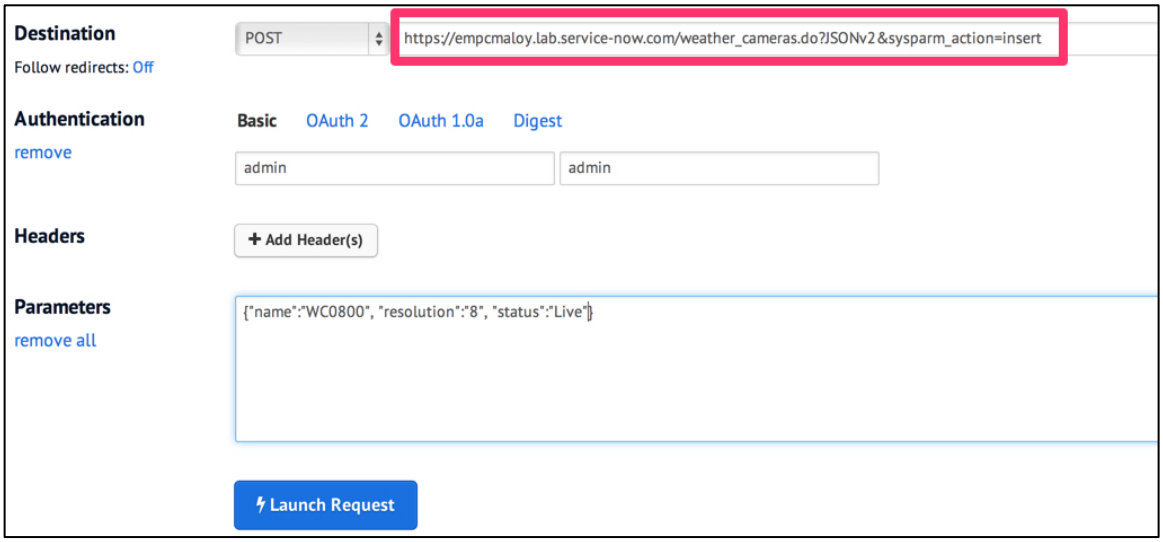
5. Open a browser tab or window to the following third party website: **http://www.hurl.it**




®2014 ServiceNow, Inc. All rights reserved.

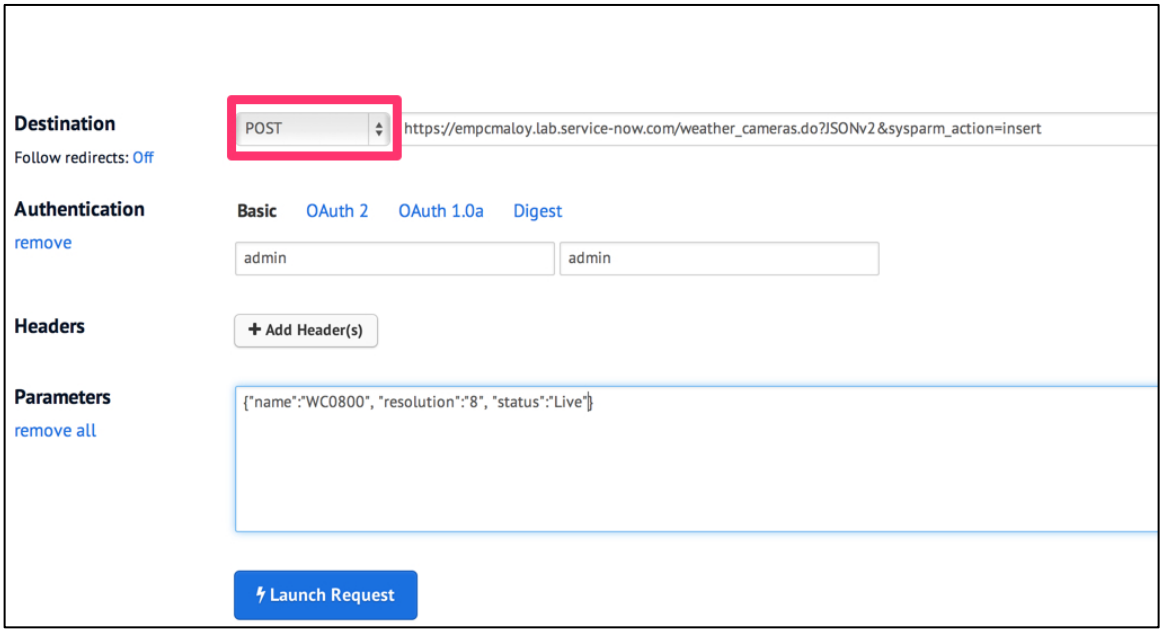
6. First, set the Destination (Replace MYINSTANCE with your own instance prefix) to **https://*MYINSTANCE*.lab.service-now.com/weather_cameras.do?JSONv2&sysparm_action=insert**

   Notice that the URL has the following URL parameters
   **JSONv2** – This triggers the JSONv2 Processor
   **sysparm_action=insert** – This tells the web service to insert a new record

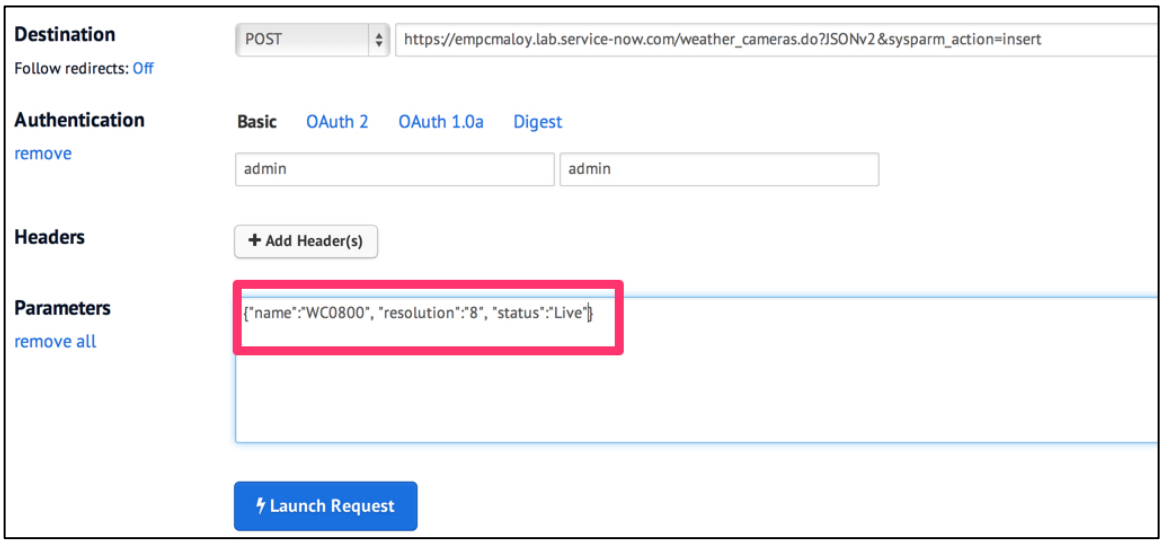



®2014 ServiceNow, Inc. All rights reserved.

7. Ensure the method select is set to **POST**.

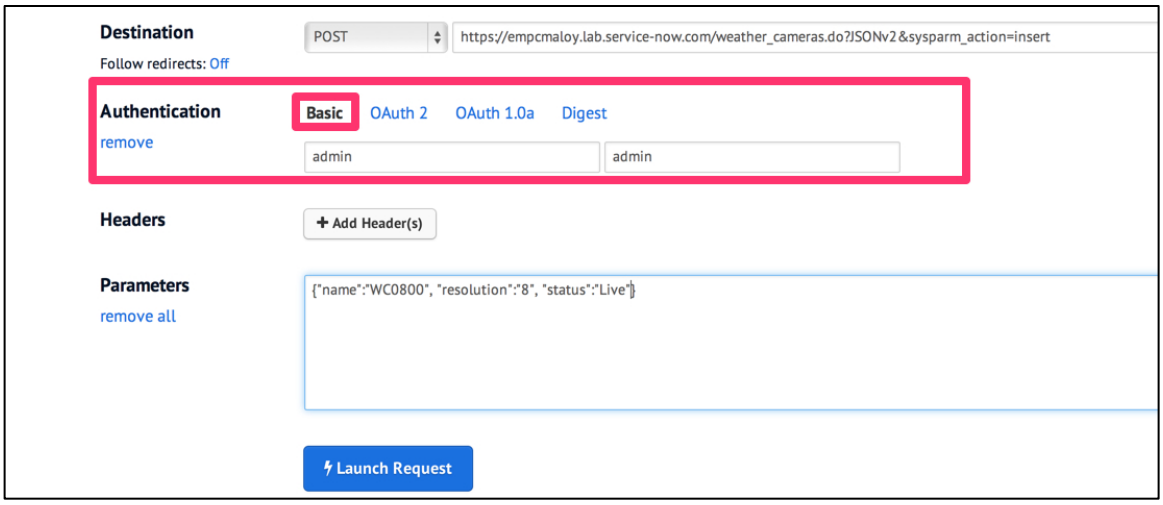



8. Set the Parameters to **{"name":"WC0800", "resolution":"8", "status":"Live"}**
This inserts a new record with those fields set to the specified values.




©2014 ServiceNow, Inc. All rights reserved.

9. Make sure the **HTTP Basic** option is select and that the proper credentials are entered, then click **Send**.



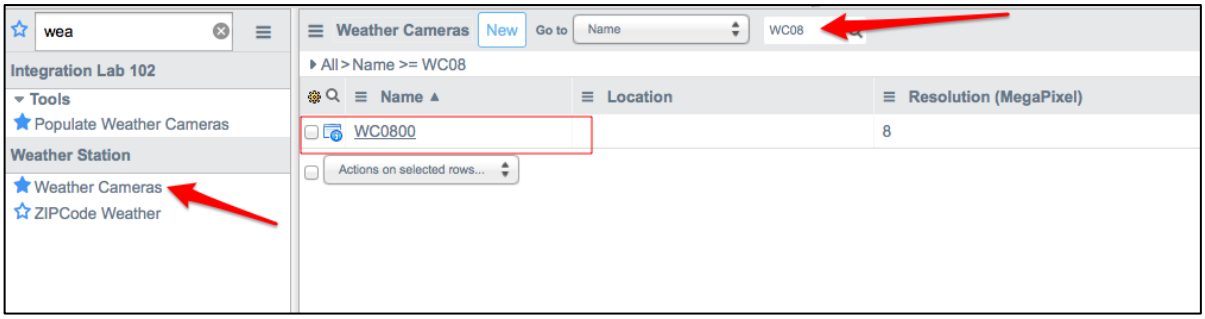

10. The response should look something similar to the screenshot below. Notice that the response contains information about the record that was created.




11. Find the **sys_id** value in the response and copy it to your clipboard for future reference.

©2014 ServiceNow, Inc. All rights reserved.

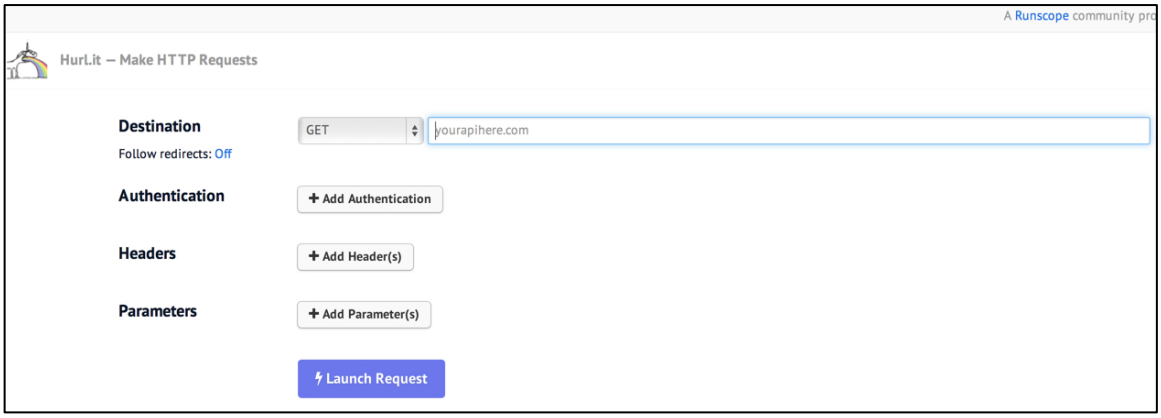
12. Verify that the camera was created by opening up a tab in your browser to ServiceNow and navigate to **Weather Station > Weather Cameras**. Filter on the camera name to ensure that the camera you created through the third party client exists.




## Use the REST client to Update the record you just created

In this exercise you use a third party REST client update the Weather Camera record that you just created. We will change the **status** to be **Down.**

1. Open a browser tab or window to the following third party website: **http://www.hurl.it**



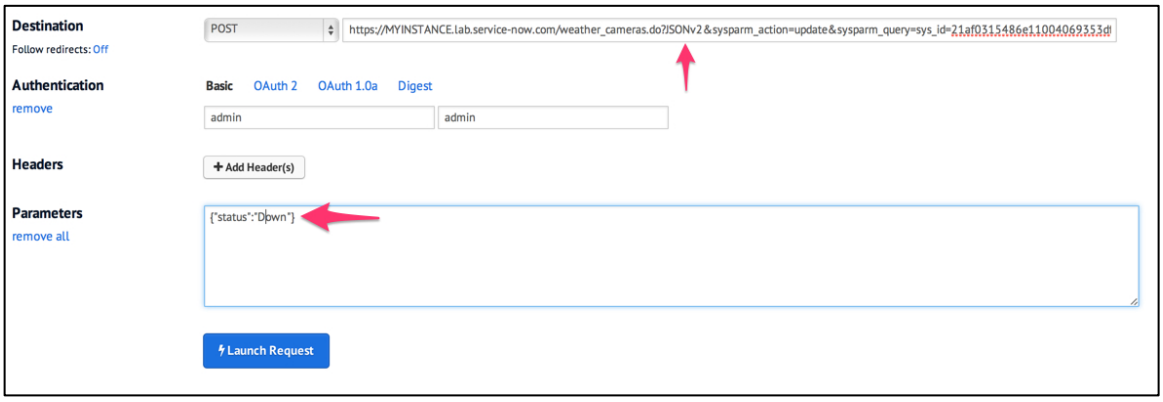

©2014 ServiceNow, Inc. All rights reserved.

2. First, set the Destination (Replace MYINSTANCE and PUT_YOUR_SYSID_HERE as appropriate) to **https://*MYINSTANCE*.lab.service-now.com/weather_cameras.do?JSONv2&sysparm_action=update&sysparm_query=sys_id=*PUT_YOUR_SYSID_HERE***

   Notice that the URL has the following URL parameters
   **JSONv2** – This triggers the JSON Web Service
   **sysparm_action=update** – This tells the web service you want to update a record
   **sysparm_query=sys_id=PUT_YOUR_SYSID_HERE** – this performs an encoded query on all the records. All matching records will be updated. In this case, it is going to only be one record since we are searching on the unique sys_id field. Remember to paste your sys_id value in this part of the URL

3. Ensure the method select is set to **POST**.

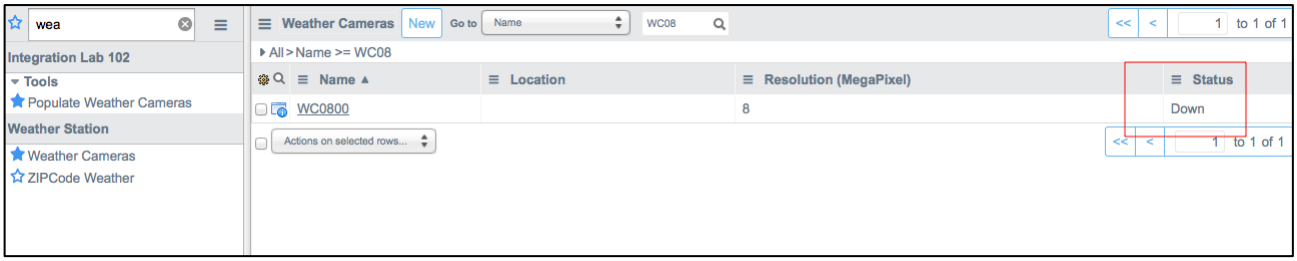
4. Set the Parameters to **{"status":"Down"}**
   This updates the matching record by setting the status field **Down**.




5. Make sure the **HTTP Basic** option is select and that the proper credentials are entered, then click **Send**.

©2014 ServiceNow, Inc. All rights reserved.

6. Verify that the camera was updated by opening up a tab in your browser to ServiceNow and navigating to **Weather Station > Weather Cameras**. Filter on the camera name and verify that the camera is now listed as **Down**.




## Query for all records that have a status of Down

In this exercise we will use a third party REST client to query the Weather Camera table for all records with a status of **Down**.

1. Open a browser tab or window to the following third party website: **http://www.hurl.it**




©2014 ServiceNow, Inc. All rights reserved.

2. Set the Destination (Replace MYINSTANCE with your own instance prefix) to **https://*MYINSTANCE*.lab.service-now.com/weather_cameras.do?JSONv2&sysparm_action=getKeys&sysparm_query=status=Down**

   Notice that the URL has the following URL parameters
   **JSONv2** – This triggers the JSON Web Service
   **sysparm_action=getKeys** – This tells the web service you want to query for sys_ids of all records that match the query
   s**ysparm_query=status=Down** – this performs an encoded query on all the records. In this case, it returns the sys_ids of all records with a status of Down

3. Ensure the method select is set to **GET**.

4. Make sure the **HTTP Basic** option is select and that the proper credentials are entered, then click **Send**.

5. Your response should be a JSON representation of an array of sys_ids:




®2014 ServiceNow, Inc. All rights reserved.

## Delete all records that have a status of "Replace"

In this exercise you use a third party REST client to delete all the Weather Camera records with a status of **Replace**.

1. Open a browser tab or window to the following third party website: **http://www.hurl.it**

2. Set the Destination (Replace MYINSTANCE with your own instance prefix) to **https://*MYINSTANCE*.lab.service-now.com/weather_cameras.do?JSONv2&sysparm_action=deleteMultiple&sysparm_query=status=Replace**

   Notice that the URL has the following URL parameters
   **JSONv2** – This triggers the JSON Web Service
   **sysparm_action=deleteMultiple** – This tells the web service you want to delete multiple record that match the query
   **sysparm_query=status=Replace** – this performs an encoded query on all the records. All records with a status of Replace will be deleted

3. Ensure the method select is set to **GET**.

4. Make sure the **HTTP Basic** option is select and that the proper credentials are entered, then click **Send**.

©2014 ServiceNow, Inc. All rights reserved.

5. Your response should be a JSON representation that shows the number of records that were affected by the deletion.




# Challenge

If you complete this lab ahead of the group, please take the time to do the following exercise(s) on your own. Feel free to review it with the instructor.

Use the standard REST Message client provided in ServiceNow to query the incident table within your instance. Have the query return the full record data for all incidents that have a priority of **2**.

©2014 ServiceNow, Inc. All rights reserved.

# Additional Resources

## ServiceNow Sites:

REST Message Wiki Documentation

http://wiki.servicenow.com/index.php?title=REST_Message

JSON Web Service Wiki Documentation

http://wiki.servicenow.com/index.php?title=JSONv2_Web_Service

## Third Party Sites:

Converting a JSON String to a JavaScript Object

http://www.john-james-andersen.com/blog/service-now/converting-a-json-string-to-an-object-in-servicenow.html

Handy tool to help formulate REST Calls to ServiceNow

http://www.hurl.it

©2014 ServiceNow, Inc. All rights reserved.