

## Web Services

HTTP-based web services allow diverse applications to talk to each other. ServiceNow supports both inbound (provider) and outbound (consumer) web services.

- **Inbound Web Services**

Inbound web services allow you to access and modify ServiceNow data using a client application.

**Direct Web Services:** Query tables and records directly using SOAP, REST, or other web service formats.

**ODBC Driver:** Report on ServiceNow data using an ODBC client, such as Microsoft Excel.

**Import Set:** access the import set tables and import data through a web service interface.

**Scripted Web Services:** define custom web service endpoints using JavaScript.

- **Outbound Web Services**

Outbound web services allow you to send SOAP and REST messages to external web service providers.

Outbound REST

Outbound SOAP

- **Direct web Service**

A direct web service is available for any table in the system provided the correct access control is setup.

The supported format of the incoming message is document style literal XML SOAP documents (Document/Literal). To retrieve the direct web service WSDL description and XML schema, point to the relative URL of <tablename>.do?WSDL. For example, to retrieve the WSDL for the Incident table on the online demo system, use the following URL:

<https://<instance name>.service-now.com/incident.do?WSDL>



### Return display value for reference variables

When you query a record using a get or getRecords function the instance returns all fields associated with that record. The fields are often reference fields that contain a sys\_id for a record on another table.

Use one of these options if you want the display value for the field to be returned instead of the sys\_id:

1. Add the property glide.soap.return\_displayValue to your system properties, and every SOAP request will return a display value for a reference field.
2. Add the parameter displayvalue=true to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field as a string, instead of the sys\_id. The SOAP URL would look as follows: <https://<instance name>.service-now.com/incident.do?displayvalue=true&SOAP>
3. Add the parameter displayvalue=all to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field, in addition to the sys\_id. The response element name for the display value field will be prefixed with dv\_ such as dv\_caller\_id.

### Retrieving journal entries using direct web services

To get the contents of a journal field, make a second soap request against the sys\_journal\_field table to pull the appropriate journal records back for the record in question. The URL for the WSDL would be in the following format

[https://instance-name.service-now.com/sys\\_journal\\_field.do?WSDL](https://instance-name.service-now.com/sys_journal_field.do?WSDL)

To retrieve the journal entries, you will first need to query the incident for its sys\_id value and then supply it as the element\_id value in a getRecordscall. To specify records only for the "comments" field, specify the value "comments" for the element field. For example, a SOAP request would look like the following.



```
<soapenv:Envelope xmlns:soapenv= "http://schemas.xmlsoap.org/soap/envelope/" xmlns:sys= "http://www.service-now.com/sys\_journal\_field" ><soapenv:Header /><soapenv:Body><sys:getRecords><element>comments</element><element_id>9d385017c611228701d22104cc95c371</element_id></sys:getRecords></soapenv:Body></soapenv:Envelope>
```

### ● Import Set Web Service

Web service import sets provides a web service interface for import set tables. By default, this type of web service will transform the incoming data synchronously based on the associated transform maps. If the associated import set mode is set to Asynchronous, the behavior is to save the data for transformation at a later time. Web service import sets tables publish all the default web service functions in the WSDL.



### System Web Services

This plugin also provides these standard import set tables:

- Computer [imp\_computer]
- Location [imp\_location]
- Notification [imp\_notification]
- User [imp\_user]

You can access web service import set WSDLs by adding .do?WSDL to the import set table URL. For example:

[http://<instance name>.service-now.com/imp\\_notification.do?WSDL](http://<instance name>.service-now.com/imp_notification.do?WSDL)

### Creating a New Web Service

Navigate to System Web Services > Create New.



← Create Web Service | = Required field
Create

Label:

Name:

Create transform map:

Target table: Incident [incident]

Create

Changes made to this list will be saved when the Create Web Service form above is saved

**Web Service Fields**
Add
☰

	Label	Name	Length
✖	<span style="border: 1px solid #ccc; padding: 2px;">Short Description</span>	<span style="border: 1px solid #ccc; padding: 2px;">u_short_description</span>	<span style="border: 1px solid #ccc; padding: 2px;">Default (40)</span>
✖	<span style="border: 1px solid #ccc; padding: 2px;">Priority</span>	<span style="border: 1px solid #ccc; padding: 2px;">u_priority</span>	<span style="border: 1px solid #ccc; padding: 2px;">Default (40)</span>

### Creating a new Web Service

The Name of the web service is the table name of the import set table whereas the Label field is the resulting table field.

If you want to create a transform map after creating the web service, check the Create transform map checkbox and choose the target table you want the data to transform into. After the Create button is clicked, the web service will be created and you will be immediately put into the Table Transform Map form. You may then continue to specify the transform map or script.

### Web Service Fields

The fields available for this web service. All fields by default are published as the XSD type of xsd:string. The Name is the field that is exposed for the web service and therefore appears as the name of the field in the WSDL. The Label is the label of the field as it appears for the import sets table.



## Mapping

During the creation of the web service import set, you can create the transform map for it. All transform maps will be run for the service when it is invoked if the import set mode is set as Synchronous.

The following image is an example of the transform map associated with the notification web service import set.

← Table Transform Map
Update Delete

Name:	SOAP notification	Created:	2008-12-17 14:39:29
Source table:	notification [soap_notification]	Target table:	Incident [incident]
Active:	<input checked="" type="checkbox"/>	Order:	100
Run business rules:	<input checked="" type="checkbox"/>	Run script:	<input checked="" type="checkbox"/>
Enforce mandatory fields:	No		
Copy empty fields:	<input type="checkbox"/>		

Script:

```
target.comments = "Timestamp: " + source.timestamp +
"\nExpires on: " + source.expires_on +
"\nDuration: " + source.duration;
```

Update Delete

### Related Links

- [Mapping Assist](#)
- [Auto map matching fields](#)

Field Maps New
◀◀ 1 to 4 of 4 ▶▶

	Source field	Target field	Coalesce
<input type="checkbox"/>	duration	calendar_duration	false
<input type="checkbox"/>	severity	severity	false
<input type="checkbox"/>	uuid	correlation_id	true
<input type="checkbox"/>	assignment_group	assignment_group	false
<input type="checkbox"/>	Actions on selected rows...		◀◀ 1 to 4 of 4 ▶▶



## SOAP transform map

- Adding Web Service Response Values

In the transform map script associated with a web service import set, setting certain variable values has the effect of changing the response values of the web service. In addition to the normal variables that are available in a transform map script, the response object holds dynamically created response elements. You can use this object to customize the response of a web service import set insert.

For example, the following code snippet inserts the `transaction_id` and `hello` variables into the response.

```
// create new elements called "transaction_id" // and "hello" in the web service  
responseresponse.transaction_id = "abc123";response.hello = "world";  
status_message="message 1"; // this is the normal status_message variable
```

This code snippet results in the following response being sent back to the web service consumer, depending on the protocol.

### SOAP

```
<soapenv:Envelope xmlns:imp="http://www.service-now.com/imp_notification"  
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">  
    <soapenv:Header/>  
    <soapenv:Body>  
        <insertResponse xmlns="http://www.service-now.com/imp_notification">  
            <sys_id>969d157c0a0a0baf008ba5770ffa798c</sys_id>  
            <table>incident</table>  
            <display_name>number</display_name>  
            <display_value>INC0010091</display_value>  
            <status>inserted</status>  
            <status_message>message 1</status_message>  
            <transaction_id>abc123</transaction_id>  
            <hello>world</hello>  
        </insertResponse>  
    </soapenv:Body></soapenv:Envelope>
```



- When displaying a mapped web service table, the following related links are available.

**Import Sets:** The import sets related to this web service import set

**Transform Maps:** A list of transform maps related to this web service

**Transform History:** The transformation history

**Edit Web Service:** Edit the web service

The following image shows a record that was inserted into the web service import set Notification. The target record is the resulting creation or modification to the Incident table record as a result of the transform.

[Notifications](#) 20 per page

---

Notifications New Go to Created

1 to 1 of 1

Created	Set	State	Target record	Message	Severity	Source	UUID	Comment
2009-01-09 16:37:17	ISET10001	Inserted	Incident: INC10008	Host 198.10.10.210 is down			HGAF76251HGF1	

Actions on selected rows... 1 to 1 of 1

### Related Links

- [Import Sets](#)
- [Transform Maps](#)
- [Transform History](#)
- [Edit Web Service](#)

## ● Scripted Web Service

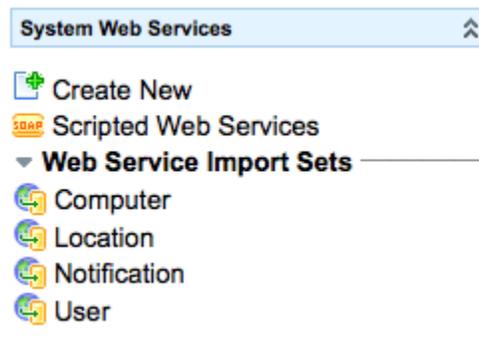
Scripted Web Services allow a ServiceNow administrator to create new web services that are not addressed by the system. You can define input and output parameters for the web service and use JavaScript to perform operations. Though this feature is very powerful, use Direct Web Services or Web Service import sets instead whenever possible since they are simpler to implement and maintain.

## ● Creating a new Web Service

When the Web Services Provider - Scripted plugin is activated, a new module Scripted



Web Services is available under the System Web Services application.



Click the module to display a list of example scripted Web Services.

sys\_web\_service  Go to

	 Name	 Active	 Short description
<input type="checkbox"/>	 <u>GetProperty</u>	true	Get a property value
<input type="checkbox"/>	 <u>GetTransactionCount</u>	true	Get the number of transactions
<input type="checkbox"/>	 <u>OrderBlackBerry</u>	true	Order a BlackBerry

Actions on selected rows...

### Example 1: Retrieving a System Property

The first step is to define the incoming and return parameters. This is done by adding an entry to the Input Parameters and Output Parameters. These parameters are used to construct and present a meaningful WSDL, and they do not add to the functionality of processing the actual Web Service itself.



**Input Parameters** [New](#) [+](#) [-](#) [Web service = GetProperty](#)

  Name
<input type="checkbox"/>  <u>property</u>
<input type="checkbox"/> Actions on selected rows... 

---

**Output Parameters** [New](#) [+](#) [-](#) [Web service = GetProperty](#)

  Name
<input type="checkbox"/>  <u>property</u>
<input type="checkbox"/> Actions on selected rows... 

The parameters are referenced in the script of the Web Service. Any of the input parameters are retrieved using the following syntax:

```
var a= request.property;
```

The output parameters are set by using the following syntax:

```
response.property = "ABC";
```

The following example demonstrates how to retrieve a system property and return it as part of the SOAP response. The example shows how to create a custom scripted Web Service to do something specific that the base ServiceNow system direct Web Services cannot.





sys_web_service		Save	Refresh	Insert	Insert and Stay
Name:	<input type="text" value="GetProperty"/>	Created:	2008-06-17 16:55:07		
Active:	<input checked="" type="checkbox"/>	Function name:	execute		
WSDL:	<a href="https://demo.service-now.com/GetProperty.do?WSDL">https://demo.service-now.com/GetProperty.do?WSDL</a>				
Short description:	Get a property value				
Script:	<pre> /*****  * Use the following business rule to invoke this example service  *  * // create the soap document  * var soapdoc = new SOAPEnvelope("GetProperty", "http://www.service-now.com/");  * soapdoc.setFunctionName("execute");  * soapdoc.addFunctionParameter("property", "glide.db.name");  *  * // post the request  * var soapRequest = new SOAPRequest("http://localhost:8080/glide/GetProperty.do?SOAP");  * var soapResponse = soapRequest.post(soapdoc);  * var property = gs.getXMLText(soapResponse, "//executeResponse/property");  *  * gs.log(property);  *  *****/  response.property = gs.getProperty(request.property); </pre>				

## ● REST Web Service

REST (REpresentational State Transfer) is a simple stateless architecture that generally runs over HTTPS/TLS. The REST style emphasizes that interactions between clients and services are enhanced by having a limited number of operations. Flexibility is provided by assigning resources their own unique universal resource indicators (URIs). Because each operation (GET, POST, PUT, and DELETE) has a specific meaning, REST avoids ambiguity.

The REST API is active by default in all instances, starting with the Eureka release.

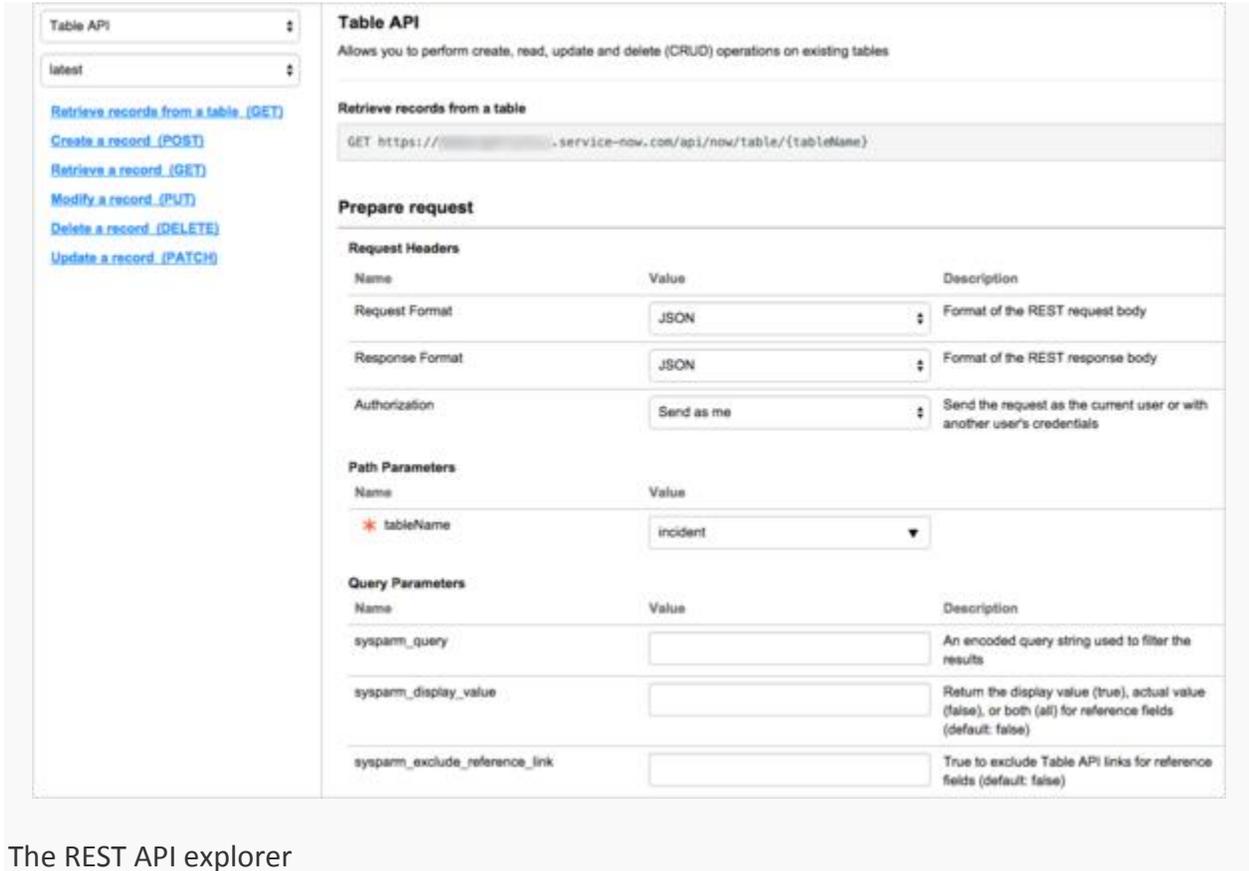
RESTful web services offer several advantages, including:

- Support for different HTTP methods to perform different actions
- Detailed response codes and header information
- Pagination support for large data sets
- Streaming data on GET requests



1. Navigate to System Web Services > REST API Explorer.

You can browse available APIs, API versions, and methods for each API.



**Table API**  
Allows you to perform create, read, update and delete (CRUD) operations on existing tables

Retrieve records from a table

GET https://...service-now.com/api/now/table/{tableName}

**Prepare request**

Name	Value	Description
Request Format	JSON	Format of the REST request body
Response Format	JSON	Format of the REST response body
Authorization	Send as me	Send the request as the current user or with another user's credentials

**Path Parameters**

Name	Value
* tableName	incident

**Query Parameters**

Name	Value	Description
sysparm_query		An encoded query string used to filter the results
sysparm_display_value		Return the display value (true), actual value (false), or both (all) for reference fields (default: false)
sysparm_exclude_reference_link		True to exclude Table API links for reference fields (default: false)

The REST API explorer

## Retrieve Existing Incidents

Use a GET request to view existing incident records.

GET https://instance.service-now.com/api/now/v1/table/incident

1. In the top-left of the REST API Explorer, select the Table API and v1 version.
2. Click Retrieve records from a table (GET).
3. In the Path Parameters section, select the Incident table.



4. Scroll to the bottom of the page and click Send.

The response includes incident records from the instance. The REST API Explorer limits queries to 10 records at a time, only the first 10 incident records appear. The response also includes a Link header that provides the URL to query the next 10 incident records.

**Response**

Status code 200 OK

**Headers**

Content-Encoding	gzip
Content-Type	application/json
Date	Tue, 06 Oct 2015 08:24:48 GMT
Link	<https://[redacted].service-now.com/api/now/table/incident?sysparm_limit=10&sysparm_offset=0>;rel="first", <https://[redacted].service-now.com/api/now/table/incident?sysparm_limit=10&sysparm_offset=10>;rel="next", <https://[redacted].service-now.com/api/now/table/incident?sysparm_limit=10&sysparm_offset=50>;rel="last"
Server	ServiceNow
Transfer-Encoding	chunked
X-Total-Count	54

**Response Body**

```
{
  "result": [
    {
      "upon_approval": "",
      "location": {
        "link": "https://[redacted].service-now.com/api/now/table/cm_location/1083361cc611227501b682158cabf646",
        "value": "1083361cc611227501b682158cabf646"
      },
      "expected_start": "",
      "reopen_count": "",
      "close_notes": "",
      "impact": "1",
      "urgency": "1",
      "correlation_id": "",
      "sys_tags": "",
      "sys_domain": {
        "link": "https://[redacted].service-now.com/api/now/table/sys_user_group/global",

```

## Create an Incident Record

Use a POST request to create a new record.

POST <https://instance.service-now.com/api/now/v1/table/incident>

1. Within the REST API Explorer, click Create a record (POST).



2. In the Path Parameters section, select the Incident table.
3. In the Request Body section, click Add a field.
4. Select a field and specify a value for that field.
5. [Optional] Click Add another field and specify additional fields to assign values to.

The request body updates automatically based on your entries, such as {"short\_description":"Test incident creation through REST", "comments":"These are my comments"}

6. After constructing the request, click Send.

The response includes a Location header that specifies where the incident was created and how to retrieve the incident. Record this header to use in the next part of this guide.



### Response

Status code	<b>201 Created</b>
-------------	--------------------

#### Headers

Content-Encoding	gzip
Content-Type	application/json
Date	Tue, 08 Sep 2015 12:11:03 GMT
Location	https://[redacted].service-now.com/api/now/v1/table/incident/ba8f71b22b1a0200b6706b7219da154f
Server	ServiceNow
Transfer-Encoding	chunked

#### Response Body

```
"short_description": "Test incident creation through REST",
"order": "",
"sys_updated_by": "admin1",
"resolved_by": "",
"notify": "1",
"upon_reject": "cancel",
"approval_history": "",
"problem_id": "",
"work_notes": "",
"calendar_duration": "",
"close_code": "",
"sys_id": "ba8f71b22b1a0200b6706b7219da154f",
```

### Read the Inserted Incident

Use the Location header from the previous POST method to run a GET request.

```
GET https://instance.service-now.com/api/now/v1/table/incident/(sys_id)
```

1. Within the REST API Explorer, click Retrieve a record (GET).
2. In the Path Parameters section, select the Incident table.
3. In the sys\_id field, enter the sys\_id of the record you created.

The record sys\_id appears as a 32-character string at the end of the POST response



Location header.

4. Click Send.

The response body contains a text representation of the record. You can control the format of the response, such as JSON or XML, using the Response Format field.

### Response

Status code	200 OK
-------------	--------

#### Headers

Content-Encoding	gzip
Content-Type	application/json
Date	Tue, 08 Sep 2015 12:13:07 GMT
Server	ServiceNow
Transfer-Encoding	chunked

#### Response Body

```
"short_description": "Test incident creation through REST",
"order": "",
"sys_updated_by": "admin1",
"resolved_by": "",
"notify": "1",
"upon_reject": "cancel",
"approval_history": "",
"problem_id": "",
"work_notes": "",
"calendar_duration": "",
"close_code": "",
"sys_id": "ba8f71b22b1a0200b6706b7219da154f",
```

### Update the Incident

You can update the incident record using either a PUT or PATCH function.



```
PUT https://instance.service-  
now.com/api/now/v1/table/incident/(sys_id)?sysparm_exclude_ref_link=true
```

1. Within the REST API Explorer, click Modify a record (PUT) or Update a record (PATCH).
2. In the Path Parameters section, select the Incident table.
3. In the sys\_id field, enter the sys\_id of the record you created.
4. In the Request Body section, click Add a field.
5. Select the short\_description field and specify a new value.
6. Click Send.
7. Verify that the response contains the updated short\_description value.

For more information on PUT and PATCH, see Table API FAQs (KB0534905).



### Response

Status code	200 OK
-------------	--------

#### Headers

Content-Encoding	gzip
Content-Type	application/json
Date	Tue, 08 Sep 2015 12:17:08 GMT
Server	ServiceNow
Transfer-Encoding	chunked

#### Response Body

```
"short_description": "This is a different short description",  
"order": "",  
"sys_updated_by": "admin1",  
"resolved_by": "",  
"notify": "1",  
"upon_reject": "cancel",  
"approval_history": "",  
"problem_id": "",  
"work_notes": "",  
"calendar_duration": "",  
"close_code": "",  
"sys_id": "ba8f71b22b1a0200b6706b7219da154f",
```

## Delete the Incident

You can delete the incident using a DELETE request.

```
DELETE https://instance.service-now.com/api/now/v1/table/incident/(sys_id)
```

1. Within the REST API Explorer, select Delete a record (DELETE).
2. In the Path Parameters section, select the Incident table.



3. In the sys\_id field, enter the sys\_id of the record you created.
4. Click Send.
5. Verify that the responses status code is 204.

### Response

Status code	<b>204 No Content</b>
-------------	-----------------------

#### Headers

Content-Encoding	gzip
Content-Type	application/json
Date	Mon, 21 Sep 2015 08:44:14 GMT
Server	ServiceNow

#### Response Body

```
""
```

