Outbound APIs

REST API

This article applies to Fuji. For more current information, see REST API [1] **at** https://developer.servicenow.com'"

The ServiceNow Wiki is no longer being updated. Please refer to https://developer.servicenow.com/app.do#!/home **for the latest information.**

Overview

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

For more information, see Getting Started with REST.

ServiceNow REST API Resources

To view the the REST API resources available in your instance, along with the correct formats for the supported methods, use the REST API Explorer. To access the explorer, navigate to **System Web Services > REST API Explorer**. (Prior to the Fuji release, navigate to **System Web Services > APIs** to view available REST APIs.)

Available APIs

The following REST APIs are available:

- **Table API:** used to perform CRUD operations on existing tables.
- **Aggregate API:** used to compute aggregate statistics about existing table and column data.
- **Import Set API:** used to provide a REST interface for import set tables.
- **Performance Analytics API:** used to query data about Performance Analytics scorecards.

Supported Methods

The following methods conform to the REST standard and are available in ServiceNow:

- GET
- POST
- PUT
- PATCH
- DELETE

Each API may support certain methods. Refer to the documentation for each API to view detailed method information.

Roles

Role name	Description
rest_service	Can use the REST API to perform REST web service operations such as querying or inserting records.
web_service_admin	Can access the System Web Services application menu and the REST modules.

API Version

REST API URIs may specify a version number, such as `/api/now/v1/table/(tablename)` or no version number, such as `/api/now/table/(tableName)`. URIs with a version number provide a consistent interface that is guaranteed to maintain backwards compatibility in future ServiceNow releases. For example, if the default response format for a particular HTTP method changes in a release, versioned URIs continue to use the previous format.

By specifying a version number in your URIs, you can ensure that future updates to the REST API do not negatively impact your integration. URIs that do not specify a version number use the most recent REST API behavior available with your instance version, which may change when you upgrade.

Headers

Each of the supported methods can be overridden if you set the `X-http-method-override` header. The `Accept` and `Content-Type` request headers are required for proper data formatting. These request headers have the following valid values:

- `Accept`: application/json, application/xml
- `Content-Type`: application/json, application/xml

POST, PUT, and PATCH operations require you to provide both headers. The GET and DELETE operations require only the `Accept` header. Failing to provide the required headers results in a **400 Bad Request** error.

Security

By default, you must include user credentials in a REST request. The REST API supports basic authentication [2] and OAuth for enforcing access controls to web resources. The user record that is used for authentication is subject to access control in the same way as an interactive user. Each request requires the proper authentication information. There is no support for mutual authentication for inbound REST requests.

REST supports cookies for binding to the existing session.

Table Access

All tables, including base system tables, global tables, and scoped tables are accessible via web services by default. You must fulfill any other web service security requirements, such as basic authentication and ACLs to access tables via web services.

You can control direct web service access to tables using the **Allow access to this table via web services** check box on the table Application Access settings. This check box must be selected to allow web service interaction with the table.

Note: *The application access fields controlling CRUD operations, such as **Can read** or **Can create** do not apply to web service requests.*

OAuth with REST

Using OAuth, you can pass a user ID and password once, and then use a token for subsequent REST requests instead of submitting credentials with each request. In this way, OAuth can improve system security by reducing the number of times you submit user credentials. You can use OAuth to authenticate REST requests starting with the Fuji release. You must enable the OAuth plugin to use this functionality.

1. Enable OAuth in the ServiceNow instance that the external client will connect to.

   You may need to activate the OAuth plugin and set the OAuth property if OAuth is not yet enabled on the instance.
2. Register a client application and create an endpoint.
3. Record the **client_id** and **client_secret** values from the previous step to use when requesting an access token.
4. To get an access token, use your REST client, such as cURL or Postman, to send a request to the OAuth endpoint (oauth_token.do).

   Format the request as a URL-encoded HTTP POST body and include the required parameters.
5. Record the access token and refresh token from the response.
6. Submit the access token with subsequent REST requests.

Click the plus to see examples of obtaining and using an OAuth access token.

In this example, the OAuth token has a **client_id** of **a329c4515612210071a5e0c298ee2be8** and a **client_secret** of **password22**.

Getting an access token

```
curl -d
"grant_type=password&client_id=a329c4515612210071a5e0c298ee2be8&client_secret=password22&username=RESTUser&password=RESTUserPassword"
https://<instance>.service-now.com/oauth_token.do
```

Sample token response

```
{
    "scope": "useraccount",
    "token_type": "Bearer",
    "expires_in": 1799,
    "refresh_token": "jZPdkEVrWvtMjrspldNjIS0uWM4D7QV9mgmcQXDVo5Qa_GVvmdR6NOp7OM038EHJnd6nZpWocFer1NcJz4zwdw",
    "access_token": "2wRlsRCT2SYjCCJP91kwo2EFzj5qg4O3I3aC09e0-0hz6Ib3YK7If-LMiNorNuglfqbkL4AfkYC92KYHUCcbpQ"
}
```

REST request with OAuth token

```
curl -H "Accept:application/json" -H "Authorization:Bearer
2wRlsRCT2SYjCCJP91kwo2EFzj5qg4O3I3aC09e0-0hz6Ib3YK7If-LMiNorNuglfqbkL4AfkYC92KYHUCcbpQ"
"https://<instance>.service-now.com/api/now/table/incident"
```

REST Response HTTP Status Codes

Status Code	Message	Details
200	Success	Success with response body.
201	Created	Success with response body.
204	Success	Success with no response body.
400	Bad Request	The request URI does not match the APIs in the system, or the operation failed for unknown reasons. Invalid headers can also cause this error.
401	Unauthorized	The user is not authorized to use the API.
403	Forbidden	The requested operation is not permitted for the user. This error can also be caused by ACL failures, or business rule or data policy constraints.
404	Not found	The requested resource was not found. This can be caused by an ACL constraint or if the resource does not exist.
405	Method not allowed	The HTTP action is not allowed for the requested REST API, or it is not supported by any API.

Dot-Walking in REST Requests

You can use dot-walking when specifying the `sysparm_query` or `sysparm_fields` parameters in requests to REST APIs that support those parameters.

Note: *The Import Set API does not support dot-walking.*

You can filter queries using related record values by dot-walking in the `sysparm_query` parameter. For example, you can retrieve all incident records where the incident **Company** has a specific **Stock symbol** value.

```
https://<instance>.service-now.com/api/now/table/incident?sysparm_query=company.stock_symbol=NYX
```

You can view field values from multiple tables by dot-walking in the `sysparm_fields` parameter. For example, you can retrieve the **Name**, **Sys_id**, and **Department** of each user that has certain roles, as well as the role **Name**.

The request runs on the User Roles [sys_user_has_role] table which defines a many-to-many relationship between users and roles. The response includes field values from the User [sys_user] and Roles [sys_user_role] tables.

```
https://<instance>.service-now.com/api/now/table/sys_user_has_role?

sysparm_fields=role%2Crole.name%2Cuser%2Cuser.name%2Cuser.sys_id%2Cuser.department&

sysparm_query=role%3D3d43716d0f6002003a2d47bce1050e0d%5EORrole%3Dac73b52d0f6002003a2d47bce1050eec&sysparm_display_value=true
```

Click the plus (+) to view a sample response including dot-walked fields

```
{
  "result": [
    {
      "user.name": "Fred Johnson",
      "user.sys_id": "f5a3716d0f6002003a2d47bce1050ed4",
      "role.name": "support",
      "user.department": {
        "display_value": "Accounting",
        "link":
"https://<instance>.service-now.com/api/now/table/cmn_department/5b3b13530f58c2003a2d47bce1050e96"
      },
      "role": {
        "display_value": "support",
        "link":
"https://<instance>.service-now.com/api/now/table/sys_user_role/3d43716d0f6002003a2d47bce1050e0d"
      },
      "user": {
        "display_value": "Fred Johnson",
        "link":
"https://<instance>.service-now.com/api/now/table/sys_user/f5a3716d0f6002003a2d47bce1050ed4"
      }
    },
    {
      "user.name": "Fred Johnson",
      "user.sys_id": "f5a3716d0f6002003a2d47bce1050ed4",
      "role.name": "asset_mgmt",
      "user.department": {
        "display_value": "Accounting",
        "link":
"https://<instance>.service-now.com/api/now/table/cmn_department/5b3b13530f58c2003a2d47bce1050e96"
      },
      "role": {
        "display_value": "asset_mgmt",
        "link":
"https://<instance>.service-now.com/api/now/table/sys_user_role/ac73b52d0f6002003a2d47bce1050eec"
      },
      "user": {
        "display_value": "Fred Johnson",
        "link":
"https://<instance>.service-now.com/api/now/table/sys_user/f5a3716d0f6002003a2d47bce1050ed4"
      }
    }
  ]
}
```

Debugging REST Queries

When the `glide.rest.debug` system property is set to **true**, it logs all REST processing, including processing durations, headers, and the request body. Prolonged use of this property can affect performance, so it is best to use it while debugging REST processing, and then set the property back to **false**.

Sample Log Output

```
2014-03-19 11:10:37 (633) http-12 New transaction
083A6031D7231100261253B2B252035C #28 /api/now/table/incident
2014-03-19 11:10:37 (653) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Started initializing REST Request
2014-03-19 11:10:37 (653) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Method:POST
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: host:localhost:8080
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: user-agent:Mozilla/5.0 (Macintosh;
Intel Mac OS X 10.7; rv:12.0) Gecko/20100101 Firefox/12.0
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: accept:application/json
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: accept-encoding:gzip, deflate
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: connection:keep-alive
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: content-type:application/json;
charset=UTF-8
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: content-length:31
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header:
cookie:glide_user_route=glide.20e7f4cd6bdc0d444810117aacc0eeae;
JSESSIONID=F07CE6ACF8AF237CB239AF43B7F360BF;
glide_user="U0N2Mjo0MDNhNjAzMWQ3MjMxMTAwMjYxMjUzYjJiMjUyMDM2OTo2ODE2Zjc5Y2MwYTgwMTY0MDFjN

glide_user_session="U0N2Mjo0MDNhNjAzMWQ3MjMxMTAwMjYxMjUzYjJiMjUyMDM2OTo2ODE2Zjc5Y2MwYTgwM
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: pragma:no-cache
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Request Header: cache-control:no-cache
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM [REST API]
RouteRegistry : Loaded Routes to Cache
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM DEBUG: [REST API]
RouteRegistry : Route loading time 0:00:00.105
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM DEBUG: [REST API]
URIHandler : Resolving URI: /now/table/incident
2014-03-19 11:10:38 (391) REST API-thread-1 SYSTEM DEBUG: [REST API]
```

```
RESTAPIProcessor : URI Resolving Duration 387570:10:38.391
2014-03-19 11:10:38 (424) REST API-thread-1 SYSTEM DEBUG: [REST API]
RESTAPIProcessor : Finished initializing REST Request
2014-03-19 11:10:38 (540) REST API-thread-1
083A6031D7231100261253B2B252035C #28 /api/now/table/incident Parameters
 -------------------------
api=api
2014-03-19 11:10:38 (541) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RESTAPIProcessor :
Processing REST Request /api/now/table/incident
2014-03-19 11:10:38 (541) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RESTAPIProcessor :
Pre-Service processing duration 0:00:00.000
2014-03-19 11:10:38 (548) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler :
Invoking Service TableAPIService
2014-03-19 11:10:38 (552) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] TableAPIService :
Inserting record
2014-03-19 11:10:38 (560) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RequestDeserializer
: Incoming Request Body:{"short_description":"test me"}
2014-03-19 11:10:39 (508) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] TableAPIService :
Glide Record Insert Duration 0:00:00.956
2014-03-19 11:10:39 (508) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] TableAPIService :
Querying for inserted record
2014-03-19 11:10:39 (513) REST API-thread-1
083A6031D7231100261253B2B252035C #### Compiler Stats ####
2014-03-19 11:10:39 (514) REST API-thread-1
083A6031D7231100261253B2B252035C Compiles: 600, time: 1,130ms
2014-03-19 11:10:39 (514) REST API-thread-1
083A6031D7231100261253B2B252035C Cache name: "syscache_expression",
max: 8,192, size: 599, seeks: 34,182, hits: 32,980, misses: 1,202,
flushed: 0
2014-03-19 11:10:39 (514) REST API-thread-1
083A6031D7231100261253B2B252035C Total classes: 1,402, bytecode length:
 3,263,462
2014-03-19 11:10:39 (514) REST API-thread-1
083A6031D7231100261253B2B252035C Total loaders created: 601, unloaded:
0, existing: 601
2014-03-19 11:10:39 (652) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] TableAPIService :
Creating service result for insert request
2014-03-19 11:10:39 (655) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler : End
```

```
 of Service InvocationTableAPIService
2014-03-19 11:10:39 (655) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler :
Service Invocation Duration 0:00:01.107
2014-03-19 11:10:39 (660) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler :
Serializing Response
2014-03-19 11:10:39 (706) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler : End
 of Response Serialization
2014-03-19 11:10:39 (706) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] ServiceHandler :
Response Serialization Duration 0:00:00.046
2014-03-19 11:10:39 (706) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RESTAPIProcessor :
Service handling duration 0:00:01.165
2014-03-19 11:10:39 (706) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RESTAPIProcessor :
End of Request Processing
2014-03-19 11:10:39 (706) REST API-thread-1
083A6031D7231100261253B2B252035C DEBUG: [REST API] RESTAPIProcessor :
REST Request Processing time 0:00:01.165
2014-03-19 11:10:39 (707) REST API-thread-1
083A6031D7231100261253B2B252035C #28 /api/now/table/incident -- total
transaction time: 0:00:02.074, total wait time: 0:00:00.001, session
wait: 0:00:00.000, semaphore wait: 0:00:00.001, source: 127.0.0.1
```

Troubleshooting

For troubleshooting information, see knowledge base article KB0535048 [3].

Enhancements

Fuji

- The REST API Explorer provides an interface for building requests to the REST API.
- You can filter GET queries using key-value pairs.
- The Aggregate API allows you to query tables and perform aggregate operations on the returned data using REST.
- The DELETE operation does not require the `Content-Type` header.
- The REST Request Timeout transaction quota prevents inbound REST requests from running for longer than 60 seconds.
- REST Requests can use OAuth type authentication.

References

[1] https://developer.servicenow.com/app.do#!/rest_api_doc?v=jakarta&id=c_TableAPI

[2] http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html#BasicAA

[3] https://hi.service-now.com/kb_view.do?sysparm_article=KB0535048

Table API

This article applies to Fuji. For more current information, see TableAPI [1] at https://developer.servicenow.com'''

The ServiceNow Wiki is no longer being updated. Please refer to the Developer Portal [1] for the latest information.

Overview

Use the ServiceNow Table API to perform create, read, update, and delete (CRUD) operations on existing tables. This API is available starting with the Eureka release.

All URIs used by the Table API are relative to `/api/`. The URIs use the following formats:

```
/api/now/table/(tableName)
/api/now/v1/table/(tableName)
/api/now/v1/table/(tableName)/(sys_id)
```

In these URIs:

`(tableName)` is the table on which the action is performed.

`(sys_id)` is the sys_id of the record on which the action is performed.

Note: *The Getting Started with REST page provides a scenario-based example for using the Table API.*

Security

Access to tables via the REST API is restricted by BasicAuth and the rest_service role. ACLs defined for tables are enforced to restrict access to data.

To allow access to tables without any authentication and authorization, add the table name to sys_public.list. ACLs defined on tables are still enforced, and it is the customer's responsibility to deactivate ACLs on tables.

Table Access

All tables, including base system tables, global tables, and scoped tables are accessible via web services by default. You must fulfill any other web service security requirements, such as basic authentication and ACLs to access tables via web services.

You can control direct web service access to tables using the **Allow access to this table via web services** check box on the table Application Access settings. This check box must be selected to allow web service interaction with the table.

Note: *The application access fields controlling CRUD operations, such as* ***Can read*** *or* ***Can create*** *do not apply to web service requests.*

Method Summary

HTTP Action + URI	Description
GET /api/now/v1/table/(tableName)	Retrieves records for the specified table.
POST /api/now/v1/table/(tableName)	Creates a record in the specified table.
GET /api/now/v1/table/(tableName)/(sys_id)	Retrieves the specified record from the specified table.
PUT /api/now/v1/table/(tableName)/(sys_id)	Updates the specified record with the request body.
PATCH /api/now/v1/table/(tableName)/(sys_id)	Updates the specified record with the request body.
DELETE /api/now/v1/table/(tableName)/(sys_id)	Deletes the specified record from the specified table.

Methods

GET /api/now/v1/table/(tableName)

This method retrieves multiple records for the specified table with proper pagination information.

Supported Parameters

Parameter	Description
sysparm_query	An encoded query. For example: `(sysparm_query=active=true)(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)` The encoded query provides support for **order by**. To sort responses based on certain fields, use the **ORDERBY** and **ORDERBYDESC** clauses in sysparm_query. For example, `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory` filters all active records and orders the results in ascending order by number first, and then in descending order by category. If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part and returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query. **Note:** *This property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.*
sysparm_display_value	Data retrieval operation for reference and choice fields. Set this parameter to one of these values: • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. **Note:** Retrieving display values may cause performance issues since the request must access fields and records on additional tables. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_fields	Comma-separated field names to return in the response.
sysparm_view	UI view to determine fields returned in the response. **Note:** If both `sysparm_fields` and `sysparm_view` are specified, the `sysparm_fields` parameter takes priority.

sysparm_limit	Limit to be applied on pagination. The default is 10000. Unusually large `sysparm_limit` values can impact system performance.
sysparm_offset	A number of records to skip before returning records. Use this parameter with `sysparm_limit` for paging results. A number of records, specified by `sysparm_offset` are skipped before starting to count the `sysparm_limit` value. For example, if `sysparm_limit` is set to 500, but there are additional records you want to query, you can specify `sysparm_offset=500` to skip the first 500 records and get the second set of records. Specifying `sysparm_offset=0` is equivalent to omitting the `sysparm_offset` parameter. Do not pass a negative number in the `sysparm_offset` parameter.
sysparm_exclude_reference_link	Additional information provided for reference fields, such as the URI to the reference resource, is suppressed.
sysparm_read_replica_category	The category value to read data from read replicas. **Note:** This parameter is not commonly used. The instance needs to have read replica support..
Key-Value Pairs	An alternative to using the `sysparm_query` parameter. You can filter a query using key-value pairs where the key is the name of a field. This functionality is available for GET queries starting with the Fuji release. For example, instead of using the parameter `&sysparm_query=active=true`, you can use `&active=true`. You can use the display value when the field is a choice or reference type field, such as `&state=closed` instead of `&state=7`. To specify multiple key-value pairs, separate each with an ampersand, such as `&active=true&assigned_to=john.smith`.

Headers

Header Type	Description
Request header	Accept

Response header	**Link:** REST message data can be split into multiple result sets rather than forcing the user to submit multiple requests. The header has different links available for the first set, previous set, next set, and the last set of records, where applicable. For example:
	`{instance}/api/now/table/cmdb_ci?sysparm_offset=40&sysparm_limit=10000>;rel="next",`
	`{instance}/api/now/table/cmdb_ci?sysparm_offset=40&sysparm_limit=10000>;rel="prev",`
	`{instance}/api/now/table/cmdb_ci?sysparm_offset=0&sysparm_limit=10000>;rel="first",`
	`{instance}/api/now/table/cmdb_ci?sysparm_offset=2780&sysparm_limit=10000>;rel="last"`
	Note: The `sysparm_limit` parameter defaults to 10,000 records. This limit can be set to any value, however, an unusually large value can impact system performance.
	X-Total-Count: The total count of records returned by query.

Note: *If no Accept header is present, the request defaults to `application/json`.*

Success Status Code

200

Language-Specific Examples for GET

- Curl
- Java
- JavaScript
- JavaScript - Server-Side
- Perl
- Python
- Ruby

JSON Response Format

Click the plus to view an example of the JSON response format

```
{
  "result":[
    {
      "upon_approval": null,
      "location": {
        "link": "http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646",
        "value": "Oklahoma"
      },
      "expected_start": "",
      "reopen_count": "",
      "close_notes": "",
      "impact": "1 - High",
      "urgency": "1 - High",
      "correlation_id": "",
      "sys_domain": {
        "link": "http://localhost:8080/api/now/table/sys_user_group/global",
        "value": "global"
      },
      "description": "User can't access email on mail.company.com.",
      "group_list": "",
      "priority": "2 - High",
      "delivery_plan": "",
      "sys_mod_count": "19",
      "work_notes_list": "",
      "follow_up": "",
      "closed_at": "2013-09-24 16:10:06",
      "sla_due": "UNKNOWN",
      "delivery_task": "",
      "sys_updated_on": "2013-10-07 18:20:56",
      "parent": "",
      "work_end": "",
      "number": "INC0000001",
      "closed_by": {
        "link": "http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0",
        "value": "Don Goodliffe"
      },
      "work_start": "",
      "calendar_stc": "86,415",
      "business_duration": "3 Hours 40 Minutes",
      "category": "Network",
      "incident_state": "Closed",
      "activity_due": "UNKNOWN",
      "correlation_display": "",
      "company": "",
      "active": "false",
      "due_date": "",
      "assignment_group": "",
      "caller_id": "",
      "knowledge": "false",
      "made_sla": "false",
      "comments_and_work_notes": "",
      "parent_incident": "",
      "state": "Closed",
      "user_input": "",
      "sys_created_on": "2012-04-24 11:24:13",
      "approval_set": "",
      "reassignment_count": "",
      "rfc": "",
      "child_incidents": "",
      "opened_at": "2013-09-23 16:09:51",
      "short_description": "Can't read email",
      "order": "",
      "sys_updated_by": "glide.maint",
      "resolved_by": "",
      "notify": "Do Not Notify",
      "upon_reject": null,
      "approval_history": "",
      "problem_id": {
        "link": "http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133",
        "value": "PRB0000007"
      },
      "work_notes": "",
      "calendar_duration": "1 Day",
      "close_code": null,
      "sys_id": "9c573169c611228700193229fff72400",
      "approval": null,
      "caused_by": "",
      "severity": "1 - High",
      "sys_created_by": "pat",
      "resolved_at": "",
      "assigned_to": {
        "link": "http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547",
        "value": "Charlie Whitherspoon"
      },
      "business_stc": "13,203",
      "cmdb_ci": {
        "link": "http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c",
        "value": "MailServerUS"
      },
      "opened_by": {
        "link": "http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7",
        "value": "Joe Employee"
      },
      "subcategory": null,
      "sys_class_name": "Incident",
      "watch_list": "",
      "time_worked": "",
      "contact_type": null,
      "escalation": "Normal",
      "comments": ""
    }
  ]
}
```

Click the plus to view an example of the JSON response format with multiple entries

```
{
  "result":[
    {
      "upon_approval": null,
      "location": {
        "link": "http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646",
        "value": "Oklahoma"
      },
      "expected_start": "",
      "reopen_count": "",
      "close_notes": "",
      "impact": "1 - High",
      "urgency": "1 - High",
      "correlation_id": "",
      "sys_domain": {
        "link": "http://localhost:8080/api/now/table/sys_user_group/global",
        "value": "global"
      },
      "description": "User can't access email on mail.company.com.",
      "group_list": "",
      "priority": "2 - High",
      "delivery_plan": "",
      "sys_mod_count": "19",
      "work_notes_list": "",
      "follow_up": "",
      "closed_at": "2013-09-24 16:10:06",
      "sla_due": "UNKNOWN",
      "delivery_task": "",
      "sys_updated_on": "2013-10-07 18:20:56",
      "parent": "",
      "work_end": "",
      "number": "INC0000001",
      "closed_by": {
        "link": "http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0",
        "value": "Don Goodliffe"
      },
      "work_start": "",
      "calendar_stc": "86,415",
      "business_duration": "3 Hours 40 Minutes",
      "category": "Network",
      "incident_state": "Closed",
      "activity_due": "UNKNOWN",
      "correlation_display": "",
      "company": "",
      "active": "false",
      "due_date": "",
      "assignment_group": "",
      "caller_id": "",
      "knowledge": "false",
      "made_sla": "false",
      "comments_and_work_notes": "",
      "parent_incident": "",
      "state": "Closed",
      "user_input": "",
      "sys_created_on": "2012-04-24 11:24:13",
      "approval_set": "",
      "reassignment_count": "",
      "rfc": "",
      "child_incidents": "",
      "opened_at": "2013-09-23 16:09:51",
      "short_description": "Can't read email",
      "order": "",
      "sys_updated_by": "glide.maint",
      "resolved_by": "",
      "notify": "Do Not Notify",
      "upon_reject": null,
      "approval_history": "",
      "problem_id": {
        "link": "http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133",
        "value": "PRB0000007"
      },
      "work_notes": "",
      "calendar_duration": "1Day",
      "close_code": null,
      "sys_id": "9c573169c611228700193229fff72400",
      "approval": null,
      "caused_by": "",
      "severity": "1 - High",
      "sys_created_by": "pat",
      "resolved_at": "",
      "assigned_to": {
        "link": "http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547",
        "value": "Charlie Whitherspoon"
      },
      "business_stc": "13,203",
      "cmdb_ci": {
        "link": "http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c",
        "value": "MailServerUS"
      },
      "opened_by": {
        "link": "http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7",
        "value": "Joe Employee"
      },
      "subcategory": null,
      "sys_class_name": "Incident",
      "watch_list": "",
      "time_worked": "",
      "contact_type": null,
      "escalation": "Normal",
      "comments": ""
    },
    {
      "upon_approval": null,
      "location": {
        "link": "http://localhost:8080/api/now/table/cmn_location/108486c7c611227500b093211aa88dcc",
        "value": "Salem OR"
      },
      "expected_start": "",
      "reopen_count": "",
      "close_notes": "",
      "impact": "1 - High",
      "urgency": "1 - High",
      "correlation_id": "",
      "sys_domain": {
        "link": "http://localhost:8080/api/now/table/sys_user_group/global",
        "value": "global"
      },
      "description": "User can't get to any of his files on the file server.",
      "group_list": "",
      "priority": "4 - Low",
      "delivery_plan": "",
      "sys_mod_count": "13",
      "work_notes_list": "",
      "follow_up": "",
      "closed_at": "",
      "sla_due": "UNKNOWN",
      "delivery_task": "",
      "sys_updated_on": "2013-10-06 17:35:37",
      "parent": "",
      "work_end": "",
      "number": "INC0000002",
      "closed_by": "",
    "work_start": "",
    "calendar_stc": "",
    "business_duration": "",
    "category": "Software",
    "incident_state": "Awaiting Problem",
    "activity_due": "UNKNOWN",
    "correlation_display": "",
    "company": "",
    "active": "true",
    "due_date": "",
    "assignment_group": "",
    "caller_id": "",
    "knowledge": "false",
    "made_sla": "false",
    "comments_and_work_notes": "",
    "parent_incident": "",
    "state": "Awaiting Problem",
    "user_input": "",
    "sys_created_on": "2012-04-24 15:30:06",
    "approval_set": "",
    "reassignment_count": "",
    "rfc": "",
    "child_incidents": "",
    "opened_at": "2013-09-28 16:07:12",
    "short_description": "Can't get to network file shares",
    "order": "",
    "sys_updated_by": "admin",
    "resolved_by": "",
    "notify": "Do Not Notify",
    "upon_reject": null,
    "approval_history": "",
    "problem_id": {
      "link": "http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133",
      "value": "PRB0000007"
    },
    "work_notes": "",
    "calendar_duration": "",
    "close_code": null,
    "sys_id": "9d385017c611228701d22104cc95c371",
    "approval": null,
    "caused_by": "",
    "severity": "1 - High",
    "sys_created_by": "pat",
    "resolved_at": "",
    "assigned_to": {
      "link": "http://localhost:8080/api/now/table/sys_user/46ca0887a9fe19810191e08e51927ebf",
      "value": "Howard Johnson"
      },
      "business_stc": "",
      "cmdb_ci": {
        "link": "http://localhost:8080/api/now/table/cmdb_ci/b0c25d1bc0a800090168be1bfcdcd759",
        "value": "FileServerFloor2"
      },
      "opened_by": {
        "link": "http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7",
        "value": "Joe Employee"
      },
      "subcategory": null,
      "sys_class_name": "Incident",
      "watch_list": "",
      "time_worked": "",
      "contact_type": null,
      "escalation": "Overdue",
      "comments": ""
    }
  ]
}
```

XML Response Format

Click the plus to view an example of the XML response format

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
   <result>
      <upon_approval>null</upon_approval>
      <location>
         <link>http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646</link>
         <value>Oklahoma</value>
      </location>
      <expected_start />
      <reopen_count />
      <close_notes />
      <impact>1 - High</impact>
      <urgency>1 - High</urgency>
      <correlation_id />
      <sys_domain>
         <link>http://localhost:8080/api/now/table/sys_user_group/global</link>
         <value>global</value>
      </sys_domain>
      <description>User can't access email on mail.company.com.</description>
      <group_list />
      <priority>2 - High</priority>
      <delivery_plan />
      <sys_mod_count>19</sys_mod_count>
      <work_notes_list />
      <follow_up />
      <closed_at>2013-09-24 16:10:06</closed_at>
      <sla_due>UNKNOWN</sla_due>
      <delivery_task />
      <sys_updated_on>2013-10-07 18:20:56</sys_updated_on>
      <parent />
      <number>INC0000001</number>
      <work_end />
      <closed_by>
         <link>http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0</link>
         <value>Don Goodliffe</value>
      </closed_by>
      <work_start />
      <calendar_stc>86,415</calendar_stc>
      <business_duration>3 Hours 40 Minutes</business_duration>
      <category>Network</category>
      <incident_state>Closed</incident_state>
      <activity_due>UNKNOWN</activity_due>
      <correlation_display />
      <company />
      <active>false</active>
      <due_date />
      <assignment_group />
      <caller_id />
      <knowledge>false</knowledge>
      <made_sla>false</made_sla>
      <comments_and_work_notes />
      <parent_incident />
      <state>Closed</state>
      <user_input />
      <sys_created_on>2012-04-24 11:24:13</sys_created_on>
      <approval_set />
      <reassignment_count />
      <rfc />
      <child_incidents />
      <opened_at>2013-09-23 16:09:51</opened_at>
      <order />
      <short_description>Can't read email</short_description>
      <resolved_by />
      <sys_updated_by>glide.maint</sys_updated_by>
      <notify>Do Not Notify</notify>
      <upon_reject>null</upon_reject>
      <approval_history />
      <problem_id>
         <link>http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133</link>
         <value>PRB0000007</value>
      </problem_id>
      <calendar_duration>1 Day</calendar_duration>
      <work_notes />
      <close_code>null</close_code>
      <approval>null</approval>
      <sys_id>9c573169c611228700193229fff72400</sys_id>
      <caused_by />
      <severity>1 - High</severity>
      <sys_created_by>pat</sys_created_by>
      <assigned_to>
         <link>http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547</link>
         <value>Charlie Whitherspoon</value>
      </assigned_to>
      <resolved_at />
      <business_stc>13,203</business_stc>
      <cmdb_ci>
         <link>http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c</link>
         <value>MailServerUS</value>
      </cmdb_ci>
      <opened_by>
         <link>http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7</link>
         <value>Joe Employee</value>
      </opened_by>
      <subcategory>null</subcategory>
      <sys_class_name>Incident</sys_class_name>
      <watch_list />
      <escalation>Normal</escalation>
      <contact_type>null</contact_type>
      <time_worked />
      <comments />
   </result>
</response>
```

Click the plus to view an example of the XML response format with multiple entries

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
   <result>
      <upon_approval>null</upon_approval>
      <location>
         <link>http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646</link>
         <value>Oklahoma</value>
      </location>
      <expected_start />
      <reopen_count />
      <close_notes />
      <impact>1 - High</impact>
      <urgency>1 - High</urgency>
      <correlation_id />
      <sys_domain>
         <link>http://localhost:8080/api/now/table/sys_user_group/global</link>
         <value>global</value>
      </sys_domain>
      <description>User can't access email on mail.company.com.</description>
      <group_list />
      <priority>2 - High</priority>
      <delivery_plan />
      <sys_mod_count>19</sys_mod_count>
      <work_notes_list />
      <follow_up />
      <closed_at>2013-09-24 16:10:06</closed_at>
      <sla_due>UNKNOWN</sla_due>
      <delivery_task />
      <sys_updated_on>2013-10-07 18:20:56</sys_updated_on>
      <parent />
      <number>INC0000001</number>
      <work_end />
      <closed_by>
         <link>http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0</link>
         <value>Don Goodliffe</value>
      </closed_by>
      <work_start />
      <calendar_stc>86,415</calendar_stc>
      <business_duration>3 Hours 40 Minutes</business_duration>
      <category>Network</category>
      <incident_state>Closed</incident_state>
      <activity_due>UNKNOWN</activity_due>
      <correlation_display />
      <company />
      <active>false</active>
      <due_date />
      <assignment_group />
      <caller_id />
      <knowledge>false</knowledge>
      <made_sla>false</made_sla>
      <comments_and_work_notes />
      <parent_incident />
      <state>Closed</state>
      <user_input />
      <sys_created_on>2012-04-24 11:24:13</sys_created_on>
      <approval_set />
      <reassignment_count />
      <rfc />
      <child_incidents />
      <opened_at>2013-09-23 16:09:51</opened_at>
      <order />
```

```
    <short_description>Can't read email</short_description>
    <resolved_by />
    <sys_updated_by>glide.maint</sys_updated_by>
    <notify>Do Not Notify</notify>
    <upon_reject>null</upon_reject>
    <approval_history />
    <problem_id>
      <link>http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133</link>
      <value>PRB0000007</value>
    </problem_id>
    <calendar_duration>1 Day</calendar_duration>
    <work_notes />
    <close_code>null</close_code>
    <approval>null</approval>
    <sys_id>9c573169c611228700193229fff72400</sys_id>
    <caused_by />
    <severity>1 - High</severity>
    <sys_created_by>pat</sys_created_by>
    <assigned_to>
      <link>http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547</link>
      <value>Charlie Whitherspoon</value>
    </assigned_to>
    <resolved_at />
    <business_stc>13,203</business_stc>
    <cmdb_ci>
      <link>http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c</link>
      <value>MailServerUS</value>
    </cmdb_ci>
    <opened_by>
      <link>http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7</link>
      <value>Joe Employee</value>
    </opened_by>
    <subcategory>null</subcategory>
    <sys_class_name>Incident</sys_class_name>
    <watch_list />
    <escalation>Normal</escalation>
    <contact_type>null</contact_type>
    <time_worked />
    <comments />
  </result>
  <result>
    <upon_approval>null</upon_approval>
    <location>
      <link>http://localhost:8080/api/now/table/cmn_location/108486c7c611227500b093211aa88dcc</link>
      <value>Salem OR</value>
    </location>
    <expected_start />
    <reopen_count />
    <close_notes />
    <impact>1 - High</impact>
    <urgency>1 - High</urgency>
    <correlation_id />
    <sys_domain>
       <link>http://localhost:8080/api/now/table/sys_user_group/global</link>
       <value>global</value>
    </sys_domain>
    <description>User can't get to any of his files on the file server.</description>
    <group_list />
    <priority>4 - Low</priority>
    <delivery_plan />
    <sys_mod_count>13</sys_mod_count>
    <work_notes_list />
    <follow_up />
    <closed_at />
    <sla_due>UNKNOWN</sla_due>
    <delivery_task />
    <sys_updated_on>2013-10-06 17:35:37</sys_updated_on>
    <parent />
    <number>INC0000002</number>
    <work_end />
    <closed_by />
    <work_start />
    <calendar_stc />
    <business_duration />
    <category>Software</category>
    <incident_state>Awaiting Problem</incident_state>
    <activity_due>UNKNOWN</activity_due>
    <correlation_display />
    <company />
    <active>true</active>
    <due_date />
    <assignment_group />
    <caller_id />
    <knowledge>false</knowledge>
    <made_sla>false</made_sla>
    <comments_and_work_notes />
    <parent_incident />
    <state>Awaiting Problem</state>
    <user_input />
    <sys_created_on>2012-04-24 15:30:06</sys_created_on>
    <approval_set />
    <reassignment_count />
    <rfc />
    <child_incidents />
```

```
      <opened_at>2013-09-28 16:07:12</opened_at>
      <order />
      <short_description>Can't get to network file shares</short_description>
      <resolved_by />
      <sys_updated_by>admin</sys_updated_by>
      <notify>Do Not Notify</notify>
      <upon_reject>null</upon_reject>
      <approval_history />
      <problem_id>
         <link>http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133</link>
         <value>PRB0000007</value>
      </problem_id>
      <calendar_duration />
      <work_notes />
      <close_code>null</close_code>
      <approval>null</approval>
      <sys_id>9d385017c611228701d22104cc95c371</sys_id>
      <caused_by />
      <severity>1 - High</severity>
      <sys_created_by>pat</sys_created_by>
      <assigned_to>
         <link>http://localhost:8080/api/now/table/sys_user/46ca0887a9fe19810191e08e51927ebf</link>
         <value>Howard Johnson</value>
      </assigned_to>
      <resolved_at />
      <business_stc />
      <cmdb_ci>         <link>http://localhost:8080/api/now/table/cmdb_ci/b0c25d1bc0a800090168be1bfcdcd759</link>
         <value>FileServerFloor2</value>
      </cmdb_ci>
      <opened_by>
         <link>http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7</link>
         <value>Joe Employee</value>
      </opened_by>
      <subcategory>null</subcategory>
      <sys_class_name>Incident</sys_class_name>
      <watch_list />
      <escalation>Overdue</escalation>
      <contact_type>null</contact_type>
      <time_worked />
      <comments />
   </result>
</response>
```

POST /api/now/v1/table/(tableName)

This method inserts one record in the specified table. Multiple record insertion is not supported by this method.

Supported Parameters

Parameter	Description
sysparm_display_value	Data retrieval operation for reference and choice fields. Based on this value, the display value and/or the actual value in the database are retrieved. • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. **Note:** There is no preferred method for setting this parameter; however, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_fields	Comma-separated field names to return in the response.
sysparm_view	UI view to determine fields returned in the response. **Note:** If both `sysparm_fields` and `sysparm_view` are specified, the `sysparm_fields` parameter takes priority.
sysparm_exclude_reference_link	Additional information provided for reference fields, such as the URI to the reference resource, is suppressed.
sysparm_input_display_value	Data insert or update operations. For values in the request: • **true** treats input values as display values and they are manipulated so they can be stored properly in the database. • **false** treats input values as actual values and stored them in the database without manipulation. **Notes:** • If this parameter is set to **true**, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format. • The preferred method is to set this parameter to **true**, but it depends on your use case. For example, if encrypted text is used, the user may not know the encrypted content, and setting this parameter to **false** might set the value in an unencrypted format in the database, which would defeat the purpose of encrypting the field. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].

Headers

Header Type	Description
Request header	**X-no-response-body:** By default, responses include body content detailing the new record. Set this header to true in the request to suppress the response body. **Content-Type:** The data format of the request body.
Response header	**Location:** The location of the created resource.

Success Status Code

201

Language-Specific Examples for POST

- Curl
- Java
- JavaScript
- JavaScript - Server-Side
- Perl
- Python
- Ruby

JSON Request Format

```
{
“field1”:”fieldvalue”,
“field2”:”fieldvalue”
}
```

XML Request Format

```
<request>
   <entry>
      <field1>fieldValue</field1>
      <field2>fieldValue</field2>
   </entry>
</request>
```

GET /api/now/v1/table/(tableName)/(sys_id)

This method retrieves the record identified by the specified sys_id from the specified table.

Note: *A 404 status code is returned if the record is not found or if the user (caller) does not have access to the record.*

Supported Parameters

Parameter	Description
sysparm_display_value	Data retrieval operation for reference and choice fields. Based on this value, the display value and/or the actual value in the database are retrieved. • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. **Note:** There is no preferred method for setting this parameter; however, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_fields	Comma-separated field names to return in the response.
sysparm_view	UI view to determine fields returned in the response. **Note:** If both `sysparm_fields` and `sysparm_view` are specified, the `sysparm_fields` parameter takes priority.
sysparm_exclude_reference_link	Additional information provided for reference fields, such as the URI to the reference resource, is suppressed.

Headers

Header Type	Description
Request Header	**Accept:** This identifies the data format in which the response should be returned.

Success Status Code

200

Language-Specific Examples for GET

- Curl
- Java
- JavaScript

- JavaScript - Server-Side
- Perl
- Python
- Ruby

JSON Response Format

Click the plus to view an example of the JSON response format

```
{
  "result":{
    "upon_approval": null,
    "location": {
      "link": "http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646",
      "value": "Oklahoma"
    },
    "expected_start": "",
    "reopen_count": "",
    "close_notes": "",
    "impact": "1 - High",
    "urgency": "1 - High",
    "correlation_id": "",
    "sys_domain": {
      "link": "http://localhost:8080/api/now/table/sys_user_group/global",
      "value": "global"
    },
    "description": "User can't access email on mail.company.com.",
    "group_list": "",
    "priority": "2 - High",
    "delivery_plan": "",
    "sys_mod_count": "19",
    "work_notes_list": "",
    "follow_up": "",
    "closed_at": "2013-09-24 16:10:06",
    "sla_due": "UNKNOWN",
    "delivery_task": "",
    "sys_updated_on": "2013-10-07 18:20:56",
    "parent": "",
    "work_end": "",
    "number": "INC0000001",
    "closed_by": {
      "link": "http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0",
      "value": "Don Goodliffe"
    },
    "work_start": "",
    "calendar_stc": "86,415",
    "business_duration": "3 Hours 40 Minutes",
    "category": "Network",
    "incident_state": "Closed",
    "activity_due": "UNKNOWN",
    "correlation_display": "",
    "company": "",
    "active": "false",
    "due_date": "",
    "assignment_group": "",
    "caller_id": "",
    "knowledge": "false",
    "made_sla": "false",
    "comments_and_work_notes": "",
    "parent_incident": "",
    "state": "Closed",
    "user_input": "",
    "sys_created_on": "2012-04-24 11:24:13",
    "approval_set": "",
    "reassignment_count": "",
    "rfc": "",
    "child_incidents": "",
    "opened_at": "2013-09-23 16:09:51",
    "short_description": "Can't read email",
    "order": "",
    "sys_updated_by": "glide.maint",
    "resolved_by": "",
    "notify": "Do Not Notify",
    "upon_reject": null,
    "approval_history": "",
    "problem_id": {
      "link": "http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133",
      "value": "PRB0000007"
    },
    "work_notes": "",
    "calendar_duration": "1 Day",
    "close_code": null,
    "sys_id": "9c573169c611228700193229fff72400",
    "approval": null,
    "caused_by": "",
    "severity": "1 - High",
    "sys_created_by": "pat",
    "resolved_at": "",
    "assigned_to": {
      "link": "http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547",
      "value": "Charlie Whitherspoon"
    },
    "business_stc": "13,203",
    "cmdb_ci": {
      "link": "http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c",
      "value": "MailServerUS"
    },
    "opened_by": {
      "link": "http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7",
      "value": "Joe Employee"
    },
    "subcategory": null,
    "sys_class_name": "Incident",
    "watch_list": "",
    "time_worked": "",
    "contact_type": null,
    "escalation": "Normal",
    "comments": ""
  }
}
```

XML Response Format

Click the plus to view an example of the XML response format

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
   <result>
      <upon_approval>null</upon_approval>
      <location>
         <link>http://localhost:8080/api/now/table/cmn_location/1083361cc611227501b682158cabf646</link>
         <value>Oklahoma</value>
      </location>
      <expected_start />
      <reopen_count />
      <close_notes />
      <impact>1 - High</impact>
      <urgency>1 - High</urgency>
      <correlation_id />
      <sys_domain>
         <link>http://localhost:8080/api/now/table/sys_user_group/global</link>
         <value>global</value>
      </sys_domain>
      <description>User can't access email on mail.company.com.</description>
      <group_list />
      <priority>2 - High</priority>
      <delivery_plan />
      <sys_mod_count>19</sys_mod_count>
      <work_notes_list />
      <follow_up />
      <closed_at>2013-09-24 16:10:06</closed_at>
      <sla_due>UNKNOWN</sla_due>
      <delivery_task />
      <sys_updated_on>2013-10-07 18:20:56</sys_updated_on>
      <parent />
      <number>INC0000001</number>
      <work_end />
      <closed_by>
         <link>http://localhost:8080/api/now/table/sys_user/9ee1b13dc6112271007f9d0efdb69cd0</link>
         <value>Don Goodliffe</value>
      </closed_by>
      <work_start />
      <calendar_stc>86,415</calendar_stc>
      <business_duration>3 Hours 40 Minutes</business_duration>
      <category>Network</category>
      <incident_state>Closed</incident_state>
      <activity_due>UNKNOWN</activity_due>
      <correlation_display />
      <company />
      <active>false</active>
      <due_date />
      <assignment_group />
      <caller_id />
      <knowledge>false</knowledge>
      <made_sla>false</made_sla>
      <comments_and_work_notes />
      <parent_incident />
      <state>Closed</state>
      <user_input />
      <sys_created_on>2012-04-24 11:24:13</sys_created_on>
      <approval_set />
      <reassignment_count />
      <rfc />
      <child_incidents />
      <opened_at>2013-09-23 16:09:51</opened_at>
      <order />
      <short_description>Can't read email</short_description>
      <resolved_by />
      <sys_updated_by>glide.maint</sys_updated_by>
      <notify>Do Not Notify</notify>
      <upon_reject>null</upon_reject>
      <approval_history />
      <problem_id>
         <link>http://localhost:8080/api/now/table/problem/9d3a266ac6112287004e37fb2ceb0133</link>
         <value>PRB0000007</value>
      </problem_id>
      <calendar_duration>1 Day</calendar_duration>
      <work_notes />
      <close_code>null</close_code>
      <approval>null</approval>
      <sys_id>9c573169c611228700193229fff72400</sys_id>
      <caused_by />
      <severity>1 - High</severity>
      <sys_created_by>pat</sys_created_by>
      <assigned_to>
         <link>http://localhost:8080/api/now/table/sys_user/46b87022a9fe198101a78787e40d7547</link>
         <value>Charlie Whitherspoon</value>
      </assigned_to>
      <resolved_at />
      <business_stc>13,203</business_stc>
      <cmdb_ci>
         <link>http://localhost:8080/api/now/table/cmdb_ci/b0c4030ac0a800090152e7a4564ca36c</link>
         <value>MailServerUS</value>
      </cmdb_ci>
      <opened_by>
         <link>http://localhost:8080/api/now/table/sys_user/681ccaf9c0a8016400b98a06818d57c7</link>
         <value>Joe Employee</value>
      </opened_by>
      <subcategory>null</subcategory>
      <sys_class_name>Incident</sys_class_name>
      <watch_list />
      <escalation>Normal</escalation>
      <contact_type>null</contact_type>
      <time_worked />
      <comments />
   </result>
</response>
```

PUT /api/now/v1/table/(tableName)/(sys_id)

This method updates the specified record with the request body.

Supported Parameters

Parameter	Description
sysparm_display_value	Data retrieval operation for reference and choice fields. Based on this value, the display value and/or the actual value in the database are retrieved. • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. **Note:** There is no preferred method for setting this parameter; however, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_fields	Comma-separated field names to return in the response.
sysparm_view	UI view to determine fields returned in the response. **Note:** If both `sysparm_fields` and `sysparm_view` are specified, the `sysparm_fields` parameter takes priority.
sysparm_exclude_reference_link	Additional information provided for reference fields, such as the URI to the reference resource, is suppressed.

sysparm_input_display_value	Data insert or update operations. For values in the request: • **true** treats input values as display values and they are manipulated so they can be stored properly in the database. • **false** treats input values as actual values and stored them in the database without manipulation. **Notes:** • If this parameter is set to **true**, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format. • The preferred method is to set this parameter to **true**, but it depends on your use case. For example, if encrypted text is used, the user may not know the encrypted content, and setting this parameter to **false** might set the value in an unencrypted format in the database, which would defeat the purpose of encrypting the field. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].

Headers

Header Type	Description
Request header	**X-no-response-body:** By default, responses include body content detailing the modified record. Set this header to true in the request to suppress the response body. **Content-Type:** The data format of the request body.

Success Status Code

200

Language-Specific Examples for PUT

- Curl
- Java
- JavaScript
- JavaScript - Server-Side
- Perl
- Python
- Ruby

JSON Request Format

```
{
“field1”:”fieldvalue”,
“field2”:”fieldvalue”
}
```

XML Request Format

```
<request>
   <entry>
      <field1>fieldValue</field1>
      <field2>fieldValue</field2>
   </entry>
</request>
```

PATCH /api/now/v1/table/(tableName)/(sys_id)

This method updates the specified record with the request body.

Supported Parameters

Parameter	Description
sysparm_display_value	Data retrieval operation for reference and choice fields. Based on this value, the display value and/or the actual value in the database are retrieved. • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. **Note:** There is no preferred method for setting this parameter; however, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_fields	Comma-separated field names to return in the response.
sysparm_view	UI view to determine fields returned in the response. **Note:** If both `sysparm_fields` and `sysparm_view` are specified, the `sysparm_fields` parameter takes priority.
sysparm_input_display_value	Data insert or update operations. For values in the request: • **true** treats input values as display values and they are manipulated so they can be stored properly in the database. • **false** treats input values as actual values and stored them in the database without manipulation. **Notes:** • If this parameter is set to **true**, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format. • The preferred method is to set this parameter to **true**, but it depends on your use case. For example, if encrypted text is used, the user may not know the encrypted content, and setting this parameter to **false** might set the value in an unencrypted format in the database, which would defeat the purpose of encrypting the field. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].

Headers

Header Type	Description
Request header	**X-no-response-body:** By default, responses include body content detailing the modified record. Set this header to true in the request to suppress the response body. **Content-Type:** The data format of the request body.

Success Status Code

200

Language-Specific Examples for PATCH

- Curl
- Java
- JavaScript
- JavaScript - Server-Side
- Perl
- Python
- Ruby

JSON Request Format

```
{
“field1”:”fieldvalue”,
“field2”:”fieldvalue”
}
```

XML Request Format

```
<request>
   <entry>
      <field1>fieldValue</field1>
      <field2>fieldValue</field2>
   </entry>
</request>
```

DELETE /api/now/v1/table/(tableName)/(sys_id)

This method deletes the specified record for the specified table.

Success Status Code

204

Language-Specific Examples for DELETE

- Curl
- Java
- JavaScript
- JavaScript - Server-Side
- Perl
- Python
- Ruby

Language Examples

The following links provide examples of Table API method usage in a variety of languages. This is a partial list of supported languages:

- Table API Curl Examples
- Table API Java Examples
- Table API JavaScript Examples
- Table API Perl Examples
- Table API Python Examples
- Table API Ruby Examples
- Table API Server-Side_JavaScript Examples

Troubleshooting

For troubleshooting information, see the Table API FAQs [2] Knowledge Base article.

References

[1] https://developer.servicenow.com/app.do#!/home
[2] https://hi.service-now.com/kb_view.do?sysparm_article=KB0534905

Import Set API

This article applies to Fuji. For more current information, see Import Set API [1] at https://developer.servicenow.com'''

The ServiceNow Wiki is no longer being updated. Please refer to https://developer.servicenow.com/app.do#!/home **for the latest information.**

Overview

The ServiceNow Import Set API provides a REST interface for import set tables. The API transforms incoming data based on associated transform maps. The import set API supports synchronous transforms.

The Import Set API mirrors the existing SOAP interface. This API is available starting with the Eureka release.

Security

Access to tables via the REST API is restricted by BasicAuth and the rest_service ACL. To allow access to tables without any authentication or authorization, add the table name to sys_public.list. ACLs defined on tables are still enforced, and it is the administrator's responsibility to deactivate ACLs them.

Table Access

All tables, including base system tables, global tables, and scoped tables are accessible via web services by default. You must fulfill any other web service security requirements, such as basic authentication and ACLs to access tables via web services.

You can control direct web service access to tables using the **Allow access to this table via web services** check box on the table Application Access settings. This check box must be selected to allow web service interaction with the table.

Note: *The application access fields controlling CRUD operations, such as* ***Can read*** *or* ***Can create*** *do not apply to web service requests.*

Roles

Certain roles are required to perform actions using the import set API.

Role	Description
import_set_loader	Allows users to load import sets.
import_scheduler	Allows users to scheduled imports.
import_transformer	Allows users to manage import set transform maps and run transforms.
import_admin	Allows users to manage all aspects of import sets and imports.

Method Summary

HTTP Action + URI	Description
POST /api/now/import/(staging TableName)	Inserts a new record into the import set staging table.
GET /api/now/import/(TableName)/(sys_id)	Retrieves a specific import set record and associated target records created as part of the transformation.

Methods

POST /api/now/import/(staging TableName)

This method inserts incoming data into a specified staging table and triggers transformation based on predefined transform maps in the import set table. Transformation occurs synchronously. For each transform map that you define, the responses include transformation results such as information on the target records.

Note: *The* `status_message` *and* `error_message` *fields on transformation scripts are processed and returned in response, along with any custom response fields.*

Request Parameters

None

Headers

Header Type	Description
Request header	**Accept:** This identifies the data format in which the response should be returned. **Content-Type:** The data format of the request body.
Response header	**Location:** The location of the created resource.

Success Status Code

201 Created

POST Examples

Click the plus to view Example 1: Loading a record to an import set

Request:

```
$ curl http://{instance}/api/now/import/imp_notification -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message", "uuid":"00004"}' -v<br>
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
>
```

Response:

```
< HTTP/1.1 201 Created
< Location:
http://{instance}/api/now/import/imp_notification/127375019f23110041a496fcc67fcfe3
< Content-Type: application/json
<
{
"result": [
{
"sys_id": "567375019f23110041a496fcc67fcfe3",
"status": "inserted",
"record_link":
"{instance}/api/now/table/incident/567375019f23110041a496fcc67fcfe3",
"display_value": "INC0010023",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Click the plus to view Example 2: Results for multiple targets are returned

Request:

```
$ curl http://{instance}/api/now/import/imp_notification -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message", "uuid":"00005"}' -v
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
>
```

Response:

```
< HTTP/1.1 201 Created
< Location:
http://{instance}/api/now/import/imp_notification/404875019f23110041a496fcc67fcfe5
{
"result": [
{
"sys_id": "484875019f23110041a496fcc67fcfe5",
"status": "inserted",
"record_link":
"http://{instance}/api/now/table/incident/484875019f23110041a496fcc67fcfe5",
"display_value": "INC0010024",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
},
{
"sys_id": "084875019f23110041a496fcc67fcfe6",
"status": "inserted",
"record_link":
"http://{instance}/api/now/table/problem/084875019f23110041a496fcc67fcfe6",
"display_value": "PRB0040025",
"display_name": "number",
"table": "problem",
"transform_map": "Problem Xform"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Click the plus to view Example 3: Using a simple transform script defined on Problem Xform

Predefined Transform Script:

```
status_message = "I’m a status message”;
```

Request:

```
$ curl http://{instance}/api/now/import/imp_notification -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message", "uuid":"00006"}' -v
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
>
```

Response:

```
< HTTP/1.1 201 Created
< Location:
http://{instance}/api/now/import/imp_notification/922975019f23110041a496fcc67fcf6a
{
"result": [
{
"sys_id": "5a2975019f23110041a496fcc67fcf6a",
"status": "inserted",
"record_link":
"record_link":
"http://{instance}/api/now/table/incident/5a2975019f23110041a496fcc67fcf6a",
"display_value": "INC0010025",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
},
{
"status_message": "I'm a status message”, <<<———————————————————— status message
"sys_id": "1a2975019f23110041a496fcc67fcf6b",
"status": "inserted",
"record_link":
"http://{instance}/api/now/table/problem/1a2975019f23110041a496fcc67fcf6b",
"display_value": "PRB0040026",
"display_name": "number",
"table": "problem",
"transform_map": "Problem Xform"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Click the plus to view Example 4: Using a script to define custom dynamic fields

Script to Define Custom Fields:

```
status_message = "I’m a status message”;
response.comments = “Comments field”;
response.another_field = “Another field”;
```

Request:

```
$ curl [2] -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message", "uuid":"00007"}' -v
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
></pre>
```

Response:

```
< HTTP/1.1 201 Created
< Location:
http://{instance}/api/now/import/imp_notification/49fa75019f23110041a496fcc67fcfe7
{
"result": [
{
"sys_id": "01fa75019f23110041a496fcc67fcfe8",
"status": "inserted",
"record_link":
"http://{instance}/api/now/table/incident/01fa75019f23110041a496fcc67fcfe8",
"display_value": "INC0010026",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
},
{
"comments": "Comments field",
"status_message": "I'm a status message",
"transform_map": "Problem Xform",
"table": "problem",
"display_name": "number",
"display_name": "number",
"display_value": "PRB0040027",
"record_link":
"http://{instance}/api/now/table/problem/cdfa75019f23110041a496fcc67fcfe8",
"status": "inserted",
"sys_id": "cdfa75019f23110041a496fcc67fcfe8",
"another_field": "Another field"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Click the plus to view Example 5: No update due to coalesce and no data change

Request:

```
$ curl http://{instance}/api/now/import/imp_notification -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message", "uuid":"00007"}' -v
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
>
```

Response:

```
< Location:
http://{instance}/api/now/import/imp_notification/e82d75019f23110041a496fcc67fcf6c
{
"result": [
{
"comments": "Comments field",
"status_message": "I'm a status message;No field values changed",
"transform_map": "Problem Xform",
"table": "problem",
"display_name": "number",
"display_value": "PRB0040027",
"record_link":
"http://{instance}/api/now/table/problem/cdfa75019f23110041a496fcc67fcfe8",
"status": "ignored",
"sys_id": "cdfa75019f23110041a496fcc67fcfe8",
"another_field": "Another field"
},
{
"status_message": "No field values changed",
"sys_id": "01fa75019f23110041a496fcc67fcfe8",
"status": "ignored",
"record_link":
"http://{instance}/api/now/table/incident/01fa75019f23110041a496fcc67fcfe8",
"display_value": "INC0010026",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Click the plus to view Example 6: Record updated

Request:

```
$ curl http://{instance}/api/now/import/imp_notification -XPOST --user admin:admin
-H "Accept: application/json" -H "Content-Type: application/json" --data
'{"message":"some message data changed", "uuid":"00007"}' -v
> POST /api/now/import/imp_notification HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
> Content-Type: application/json
>
```

Response:

```
< HTTP/1.1 201 Created
< Location:
http://{instance}/api/now/import/imp_notification/dedd75019f23110041a496fcc67fcfe9
{
"result": [
{
"comments": "Comments field",
"status_message": "I'm a status message;No field values changed",
"transform_map": "Problem Xform",
"table": "problem",
"display_name": "number",
"display_value": "PRB0040027",
"record_link":
"http://{instance}/api/now/table/problem/cdfa75019f23110041a496fcc67fcfe8",
"status": "ignored",
"sys_id": "cdfa75019f23110041a496fcc67fcfe8",
"another_field": "Another field"
},
{
"sys_id": "01fa75019f23110041a496fcc67fcfe8",
"status": "updated",
"record_link":
"http://{instance}/api/now/table/incident/01fa75019f23110041a496fcc67fcfe8",
"display_value": "INC0010026",
"display_name": "number",
"table": "incident",
"transform_map": "Notification"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

GET /api/now/import/(tableName)/(sys_id)

This method retrieves the associated record and resulting transformation result. As import set tables are deleted frequently based on a schedule, GET operations may return `404 NotFound` responses if the transformation result no longer exists.

Supported Parameters

None

Headers

Header Type	Description
Request Header	**Accept:** This identifies the data format in which the response should be returned.

Success Status Code

200

GET Example

Click the plus to view this GET Example: Getting the results again using the Location header link

Request:

```
 $ curl
http://{instance}/api/now/import/imp_notification/127375019f23110041a496fcc67fcfe3
-XGET --user admin:admin -H "Accept: application/json" -v<br>
> GET /api/now/import/imp_notification/127375019f23110041a496fcc67fcfe3 HTTP/1.1
> Authorization: Basic YWRtaW46YWRtaW4=
> Accept: application/json
>
```

Response:

```
< HTTP/1.1 200 OK
<
{
"result": [
{
"sys_id": "567375019f23110041a496fcc67fcfe3",
"status": "inserted",
"record_link":
"http://{instance}/api/now/table/incident/567375019f23110041a496fcc67fcfe3",
"display_value": "INC0010023",
"display_name": "number",
"table": "incident",
"table": "incident",
"transform_map": "Notification"
}
],
"staging_table": "imp_notification",
"import_set": "ISET0010004"
}
```

Note: *This example uses the same record described in the POST examples.*

Troubleshooting

For general information on troubleshooting the performance of import set jobs, see Troubleshooting Import Set Performance.

For FAQs on using REST, see the knowledge base article KB0535048 [3].

References

[1] https://developer.servicenow.com/app.do#!/rest_api_doc?v=jakarta&id=c_ImportSetAPI
[2] http://{instance}/api/now/import/imp_notification

Aggregate API

This article applies to Fuji. For more current information, see Aggregate API [1] **at** https:/ / developer.servicenow.com"'

The ServiceNow Wiki is no longer being updated. Please refer to the Developer Portal [1] **for the latest information.**

Overview

The ServiceNow Aggregate REST API allows you to compute aggregate statistics about existing table and column data. The Aggregate API is available starting with the Fuji release.

All URIs used by the Aggregate API are relative to /api/. The URIs use the following format:

```
/api/now/stats/(tableName)
/api/now/v1/stats/(tableName)
```

In these URIs, (tableName) is the table on which the action is performed.

Note: *This URI differs from the Table API URI, using **stats** instead of **table**.*

Security

Access to tables via the REST API is restricted by BasicAuth and the rest_service role. ACLs defined for tables are enforced to restrict access to data.

To allow access to tables without any authentication and authorization, add the table name to sys_public.list. ACLs defined on tables are still enforced, and it is the customer's responsibility to deactivate ACLs on tables.

Table Access

All tables, including base system tables, global tables, and scoped tables are accessible via web services by default. You must fulfill any other web service security requirements, such as basic authentication and ACLs to access tables via web services.

You can control direct web service access to tables using the **Allow access to this table via web services** check box on the table Application Access settings. This check box must be selected to allow web service interaction with the table.

Note: *The application access fields controlling CRUD operations, such as **Can read** or **Can create** do not apply to web service requests.*

Method Summary

The Aggregate API supports only the GET action. Aggregate queries never update records.

HTTP Action + URI	Description
GET /api/now/v1/stats/(tableName)	Retrieves records for the specified table and performs aggregate functions on the returned values.

Methods

GET /api/now/v1/stats/(tableName)

Supported Parameters

Parameter	Description
sysparm_query	An encoded query. For example: `(sysparm_query=active=true)(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)` If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part and returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query. **Note:** *This property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.*
sysparm_group_by	The fields to group the returned data by. You can specify multiple fields by separating each with a comma, such as `sysparm_group_by=priority,state`.
sysparm_having	An additional query allowing you to filter the data based on an aggregate operation. The value for this parameter must follow the syntax `aggregate^field^operator^value`, such as `count^priority^>^3` to obtain the number of records within the query results with a priority greater than 3. You can specify multiple queries by separating each with a comma, such as `count^state^=^1,avg^priority^>^3`. For more information, see Available Aggregate Functions.
sysparm_<aggregate>_fields	The list of fields you want to perform each aggregate operation on. You can specify multiple fields by separating each with a comma. For example, to get the average values from the duration and priority fields, use `sysparm_avg_fields=duration,priority`. For more information, see Available Aggregate Functions. **Note:** *You must specify this parameter, the `sysparm_count` parameter, or both for your query to return meaningful results. If neither parameter is passed, no aggregate operation is performed.*
sysparm_count	A boolean flag. You can set this parameter to **true** for the number of records returned by the query. **Note:** *You must specify this parameter, the `sysparm_<aggregate>_fields` parameter, or both for your query to return meaningful results. If neither parameter is passed, no aggregate operation is performed.*

sysparm_display_value	Data retrieval operation when grouping by reference or choice fields. Based on this value, the query returns either the display value, the actual value in the database, or both. • **true** returns display values for all of the fields. • **false** returns actual values from the database. If a value is not specified, this parameter defaults to **false**. • **all** returns both actual and display values. There is no preferred method for setting this parameter; however, specifying the display value may cause performance issues because it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905) [2].
sysparm_order_by	A list of values to order grouped results by. You can specify an order using a field or an aggregate. For example, if you specify `sysparm_order_by=AVG^state`, groups of results with lower average state values are returned first. You can also order by COUNT to arrange groups of records by the number of records in each group. When you specify an order, groups are ordered in ascending order by default. Use **^DESC** to sort in descending order, such as `sysparm_order_by=state^DESC`.
Key-Value Pairs	An alternative to using the `sysparm_query` parameter. You can filter a query using key-value pairs where the key is the name of a field. This functionality is available for GET queries starting with the Fuji release. For example, instead of using the parameter `&sysparm_query=active=true`, you can use `&active=true`. You can use the display value when the field is a choice or reference type field, such as `&state=closed` instead of `&state=7`. To specify multiple key-value pairs, separate each with an ampersand, such as `&active=true&assigned_to=john.smith`.

Headers

The Aggregate API uses the standard REST headers.

Success Status Code

200

For examples of queries and the available response formats, see Aggregate API Examples.

Available Aggregate Functions

You can specify which aggregate functions to perform by using either the `sysparm_<aggregate>_fields` parameter or `sysparm_having=<aggregate>^field^operator^value` parameter, substituting <aggregate> for one of these aggregate functions:

- avg
- sum
- min
- max

References

[1] https://developer.servicenow.com/app.do#!/rest_api_doc?v=jakarta&id=r_AggregateAPI-GET

Aggregate API Examples

This article applies to Fuji. For more current information, see Aggregate API [1] at https:/ / developer. servicenow.com'''

The ServiceNow Wiki is no longer being updated. Please refer to the Developer Portal [1] for the latest information.

Overview

You can use the Aggregate API to query tables and perform aggregate operations on the returned data. These examples demonstrate how to perform a REST query using cURL commands, and the data returned for each command.

Sample Response Formats

You can specify the data format as JSON or XML using the request headers.

JSON Response Format

```
{
    "result": {
        "stats": {
            "count": "6",
            "max": {
                "number": "INC0000053"
            },
            "min": {
                "priority": "1",
                "number": "INC0000015"
            }
        }
    }
}
```

XML Response Format

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
   <result>
      <stats>
         <min>
            <priority>1</priority>
            <number>INC0000007</number>
         </min>
         <max>
            <number>INC0010004</number>
         </max>
         <count>16</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value />
      </groupby_fields>
   </result>
   <result>
      <stats>
         <min>
            <priority>4</priority>
            <number>INC0000010</number>
         </min>
         <max>
            <number>INC0000012</number>
         </max>
         <count>2</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value>Database</value>
      </groupby_fields>
   </result>
   <result>
      <stats>
         <min>
            <priority>1</priority>
            <number>INC0000005</number>
         </min>
         <max>
            <number>INC0000050</number>
         </max>
         <count>9</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value>Hardware</value>
      </groupby_fields>
   </result>
   <result>
      <stats>
         <min>
            <priority>1</priority>
            <number>INC0000002</number>
         </min>
         <max>
            <number>INC0000039</number>
         </max>
         <count>5</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value>Network</value>
      </groupby_fields>
   </result>
   <result>
      <stats>
         <min>
            <priority>1</priority>
            <number>INC0000001</number>
         </min>
         <max>
            <number>INC0000055</number>
         </max>
         <count>12</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value>Service Desk</value>
      </groupby_fields>
   </result>
   <result>
      <stats>
         <min>
            <priority>1</priority>
            <number>INC0000006</number>
         </min>
         <max>
            <number>INC0000053</number>
         </max>
         <count>9</count>
      </stats>
      <groupby_fields>
         <field>assignment_group</field>
         <value>Software</value>
      </groupby_fields>
   </result>
</response>
```

Specifying a Query

You can specify a query by using either the `sysparm_query` parameter or key-value pairs.

Using sysparm_query:

```
curl -H "Accept:application/json" --user admin:admin
"http://<instance>.service-now.com/api/now/stats/incident?sysparm_count=true&sysparm_max_fields=number&sysparm_min
```

Using key-value pairs:

```
curl -H "Accept:application/json" --user admin:admin
"http://<instance>.service-now.com/api/now/stats/incident?sysparm_count=true&sysparm_max_fields=number&sysparm_min
```

```
{
    "result": {
        "stats": {
            "count": "6",
            "max": {
                "number": "INC0000053"
            },
            "min": {
                "priority": "1",
                "number": "INC0000015"
            }
        }
    }
}
```

Grouping Results

You can group results by using the `sysparm_group_by` parameter.

```
curl -H "Accept:application/json" --user admin:admin
"http://<instance>.service-now.com/api/now/stats/incident?sysparm_count=true&sysparm_max_fields=number&sysparm_min
```

```
{
    "result": [
        {
            "stats": {
                "count": "13",
                "max": {
                    "number": "INC0000048"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000007"
                }
            },
            "groupby_fields": [
                {
                    "value": "",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "2",
                "max": {
                    "number": "INC0000012"
                },
                "min": {
                    "priority": "4",
                    "number": "INC0000010"
                }
            },
            "groupby_fields": [
                {
                    "value": "287ee6fea9fe198100ada7950d0b1b73",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "9",
                "max": {
                    "number": "INC0000050"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000005"
                }
            },
            "groupby_fields": [
                {
                    "value": "8a5055c9c61122780043563ef53438e3",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "5",
                "max": {
                    "number": "INC0000039"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000002"
                }
            },
            "groupby_fields": [
                {
                    "value": "287ebd7da9fe198100f92cc8d1d2154e",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "12",
                "max": {
                    "number": "INC0000055"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000001"
                }
            },
            "groupby_fields": [
                {
                    "value": "d625dccec0a8016700a222a0f7900d06",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "9",
                "max": {
                    "number": "INC0000053"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000006"
                }
            },
            "groupby_fields": [
                {
                    "value": "8a4dde73c6112278017a6a4baf547aa7",
                    "field": "assignment_group"
                }
            ]
        }
    ]
```

```
}
```

Specifying a Display Value Format

You can control if reference and choice field values appear as a sys_id, a label value, or both by using the `sysparm_display_value` parameter.

curl -H "Accept:application/json" --user admin:admin "http://<instance>.service-now.com/api/now/stats/incident?sysparm_count=true&sysparm_max_fields=number&sysparm_min

```
{
    "result": [
        {
            "stats": {
                "count": "13",
                "max": {
                    "number": "INC0000048"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000007"
                }
            },
            "groupby_fields": [
                {
                    "value": "",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "2",
                "max": {
                    "number": "INC0000012"
                },
                "min": {
                    "priority": "4",
                    "number": "INC0000010"
                }
            },
            "groupby_fields": [
                {
                    "value": "Database",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "9",
                "max": {
                    "number": "INC0000050"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000005"
                }
            },
            "groupby_fields": [
                {
                    "value": "Hardware",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "5",
                "max": {
                    "number": "INC0000039"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000002"
                }
            },
            "groupby_fields": [
                {
                    "value": "Network",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "12",
                "max": {
                    "number": "INC0000055"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000001"
                }
            },
            "groupby_fields": [
                {
                    "value": "Service Desk",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "9",
                "max": {
                    "number": "INC0000053"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000006"
                }
            },
            "groupby_fields": [
                {
                    "value": "Software",
                    "field": "assignment_group"
                }
            ]
        }
    ]
}
```

Ordering Grouped Fields

You can order groups of results based on the values in one or more specified fields by using the `sysparm_order_by` parameter.

curl -H "Accept:application/json" --user admin:admin "http://<instance>.service-now.com/api/now/stats/incident?sysparm_count=true&sysparm_max_fields=number&sysparm_min

```
{
    "result": [
        {
            "stats": {
                "count": "6",
                "max": {
                    "number": "INC0000053"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000015"
                }
            },
            "groupby_fields": [
                {
                    "value": "Software",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "5",
                "max": {
                    "number": "INC0000055"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000017"
                }
            },
            "groupby_fields": [
                {
                    "value": "Service Desk",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "3",
                "max": {
                    "number": "INC0000039"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000002"
                }
            },
            "groupby_fields": [
                {
                    "value": "Network",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "4",
                "max": {
                    "number": "INC0000050"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000005"
                }
            },
            "groupby_fields": [
                {
                    "value": "Hardware",
                    "field": "assignment_group"
                }
            ]
        },
        {
            "stats": {
                "count": "12",
                "max": {
                    "number": "INC0000048"
                },
                "min": {
                    "priority": "1",
                    "number": "INC0000007"
                }
            },
            "groupby_fields": [
                {
                    "value": "",
                    "field": "assignment_group"
                }
            ]
        }
    ]
}
```

Outbound REST Web Service

Note: *This article applies to Fuji. For more current information, see Outbound REST Web Service* [1] *at* http://docs.servicenow.com The ServiceNow Wiki is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

Overview

ServiceNow outbound REST functionality allows you to retrieve, create, update, or delete data on a web server that supports the REST architecture [2]. You can send REST messages from a MID Server, which allows REST to access an internal network. You can send REST messages using scripts or a workflow activity to retrieve or manage data on a REST endpoint, and then use the returned response in your ServiceNow instance.

ServiceNow REST functionality is flexible enough to accommodate many web service APIs. Be sure you are familiar with your web service and the parameters it accepts before attempting to define a REST message in ServiceNow. The examples used here are configured for a specific web service API and are not intended to represent your environment.

Note: *The outbound REST web service UI and script API are significantly enhanced with the Fuji release. This page describes the current behavior. For previous version information, see Outbound REST Web Service - Versions Prior to Fuji.*

Video Tutorial

The following video tutorial demonstrates how to configure outbound REST web service messages to consume third-party web services from the ServiceNow platform. Applies to all supported releases as of Fuji.

Web Services: How to Consume Third-Party REST Web Services

REST Web Service Example

View the REST web service configuration used for the examples on this page.

This REST-based web service manages very simple records of offices and users at a company and is configured as follows.

Element	Description
Endpoint	The general endpoint for this web service is **http://10.0.103.202** with the user records at **http://10.0.103.202/users** and the office records at **http://10.0.103.202/offices**.
Authentication	The web service supports HTTP basic authentication.
GET	The HTTP GET method retrieves a single record or the entire collection. A single record can be specified with the endpoint URL or with a parameter. For example, if a user record has an ID of 12, the URL method for this web service would be **http://10.0.103.202/users/12**, and the parameter version would be **http://10.0.103.202/users?id=12**. The endpoint for the entire collection would be **http://10.0.103.202/users**. For the **offices** collection, replace the word **users** with **offices**. If you pass an HTTP parameter of `type=xml`, the data returned is in the XML format, otherwise the GET returns the data in JSON format.
POST	The HTTP POST method adds a new record to the collection. POST expects the content-type header to be `application/json`, and the content being posted to use the following format: • **users**: `{"name":"<the user's name>","age":"<the user's age>"}` • **offices**: `{"city":"<the office location>","state":"${<the state>}","zip":"<the office's zip code>"}` POST returns the ID of the newly added record in the **Response** body.
PUT	The HTTP PUT method updates an existing record. PUT takes content data in the same format as POST and uses the ID for the record being updated as specified in the single record version of GET.
DELETE	The HTTP DELETE method removes an existing record from a collection. DELETE uses the ID for the record being deleted as specified in the single record version of GET.

REST Message

ServiceNow sends requests to a REST endpoint using a REST message. A REST message contains the following elements:

- **Endpoint:** The URL of the data to be retrieved, updated, or deleted. The endpoint can be defined at the top level of the message, at the method level, or in the REST message workflow activity. The endpoint configured in the activity takes precedence over the other endpoints defined. The examples on this page show two collections: `/users` and `/office`, which were created in the example web service.
- **Headers:** The HTTP headers in REST messages carry the information about the data format. They apply to all methods in that message unless the header is overridden in the method definitions. Headers cannot be set in the REST message workflow activity.
- **Methods:** The HTTP methods—**PUT**, **POST**, **GET**, **DELETE**—interact with the data at the endpoint. Override the endpoint or header defined in the REST message by specifying alternative values in the method definitions. Each REST Message workflow activity calls a single method to perform work on the endpoint.

Configuring a REST Message

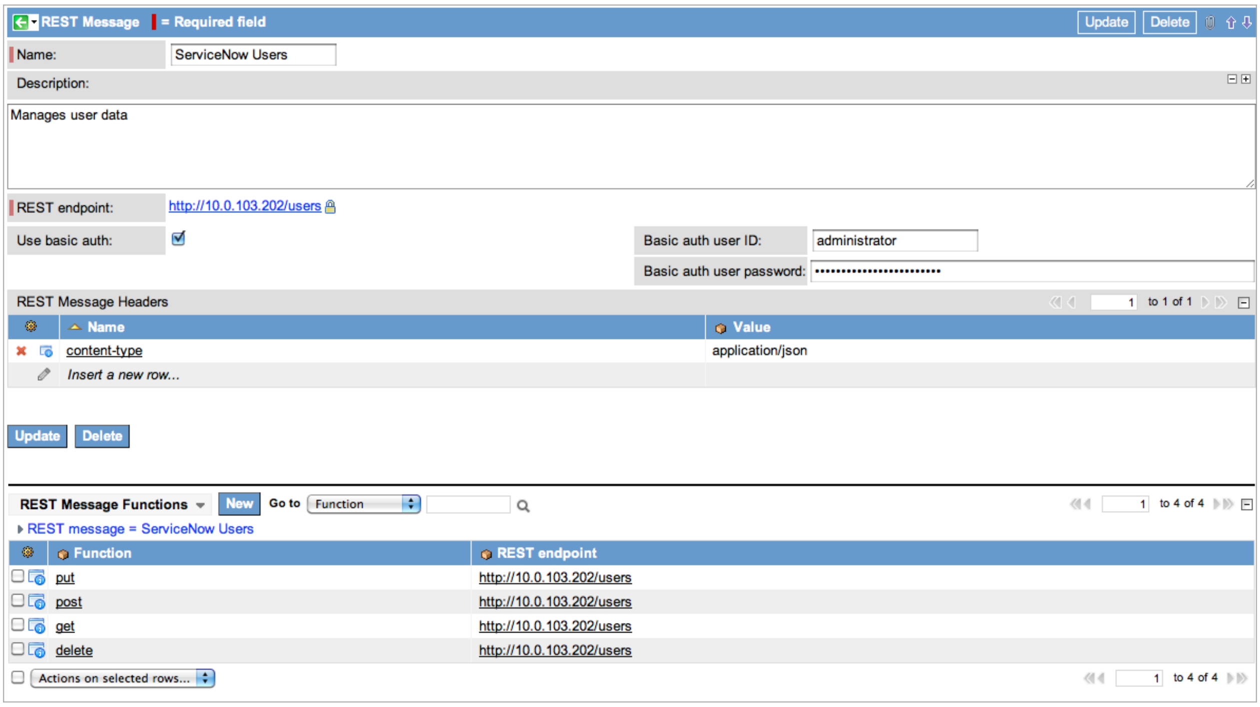
1. Navigate to **System Web Services > Outbound > REST Message**.
2. Click **New**.
3. Complete the following fields and lists:
    - **Name:** enter a unique and descriptive name for this message.
    - **Description:** enter a description of this message.
    - **Endpoint:** enter the endpoint that applies to all the methods for this REST message. This is a required field, but the endpoint configured here can be overridden at the method level.
    - **Use basic authentication:** select this check box to display the user name and password fields for configuring basic authentication for this message. These credentials are included in the message header. Authentication

configured here is inherited by the methods when the new record is submitted. If basic authentication is changed after the HTTP method record is created, the method *does not* inherit the updated authentication settings. You can configure basic authentication for each method, which overrides any authentication setting at the message level. See HTTP Methods for additional details about authentication at the method level.

- **HTTP Headers:** double-click a row in this related list to define the name-value pair for the header in this message. See List of HTTP Header Fields [3] for a list of HTTP header fields.

4. Click **Submit**.

   The system saves the record and automatically creates the methods (**GET**, **PUT**, **POST**, and **DELETE**).



Name:	ServiceNow Users
Description:	Manages user data
REST endpoint:	http://10.0.103.202/users
Use basic auth:	☑
Basic auth user ID:	administrator
Basic auth user password:	••••••••••••••••••••••••

REST Message Headers

1 to 1 of 1

Name	Value
content-type	application/json
Insert a new row...	

REST Message Functions

1 to 4 of 4

REST message = ServiceNow Users

	Function	REST endpoint
□	put	http://10.0.103.202/users
□	post	http://10.0.103.202/users
□	get	http://10.0.103.202/users
□	delete	http://10.0.103.202/users

1 to 4 of 4

HTTP Methods

HTTP methods perform the data requests that update, create, retrieve, or delete data at the endpoint with a REST message.

When you create a REST message record, several default HTTP methods are automatically created using settings inherited from the REST message record, such as the Endpoint. Subsequent changes to the REST message record are not applied to the HTTP methods automatically. You can create additional HTTP methods or modify the default HTTP methods to implement new behavior.

This example uses PUT to add data to a different REST endpoint than that defined in the parent REST message and then returns that data with GET. The concepts shown in this example apply to all method definitions. The parameters used in these procedures were configured for the web service described under REST Web Service Example.

1. Navigate to **System Web Services > Outbound > REST Message**.
2. Open the appropriate REST message and select the **PUT** method.

   The PUT method record displays the settings of the parent message when it is first opened.

3. Click the lock icon in the **Endpoint** field and edit the method to override the endpoint, replacing `/users` with `/offices`.

   The **HTTP Methods** related list in the parent REST Message record shows any changes to the endpoint made at the method level.

4. Leave the following fields blank:

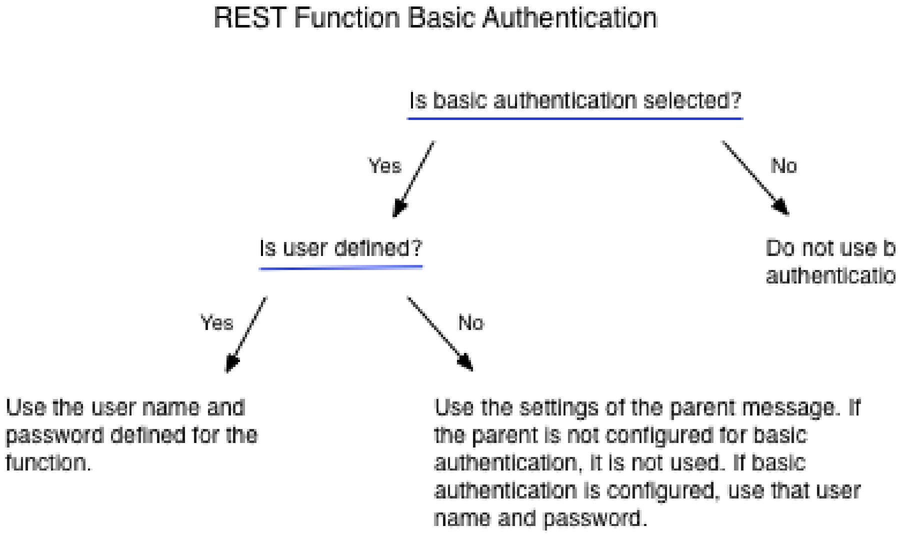
   - **Name** and **Value** in the the **HTTP Headers** embedded list

- **Basic authentication user ID**
- **Basic authentication password**

  The method uses the values configured for those elements in the parent REST Message record.

5. Select the **Use basic authentication** check box.

   This selection forces the method to use the basic authentication configured in the parent record. The diagram describes how basic authentication is determined for REST messages.


REST Function Basic Authentication
Is basic authentication selected?
Yes
No
Is user defined?
Do not use basic authentication.
Yes
No
Use the user name and password defined for the function.
Use the settings of the parent message. If the parent is not configured for basic authentication, it is not used. If basic authentication is configured, use that user name and password.

6. Select a MID Server to use for this method.

   Using a MID Server for REST messages enables an administrator to make data requests from an instance to a web services provider within a protected, local network.

7. If the REST endpoint accepts parameters, create a parameter definition in the **HTTP Query Parameters** embedded list.

   Parameters are available for all methods and can be used for a variety of operations. The REST message includes all values that match the definitions provided in the **Order** specified. This PUT uses a parameter where the **Name** is **id** and the **Value** is **7** from the sample web service.

8. Place the following statement in the **Content** field to pass content into the request. The **Content** field is only available for the POST and PUT methods.

   ```
   {"city":"${city}","state":"${state}","zip":"${zip}"}
   ```

   The format for the content must match the format expected by the web service. In this example, the web service expects **city**, **state**, and **zip** in JSON format for the `/offices` endpoint.

The method is now configured. You can test the request to ensure the behavior is as expected.

Sending a Request Through a MID Server

You can configure an HTTP method to be sent through a MID Server. By using a MID Server, the request can reach an endpoint that is behind a firewall or within a private network. To configure an HTTP method to use a MID Server, select a MID Server in the **Use MID Server** field. The instance must have an active MID Server to use this functionality.

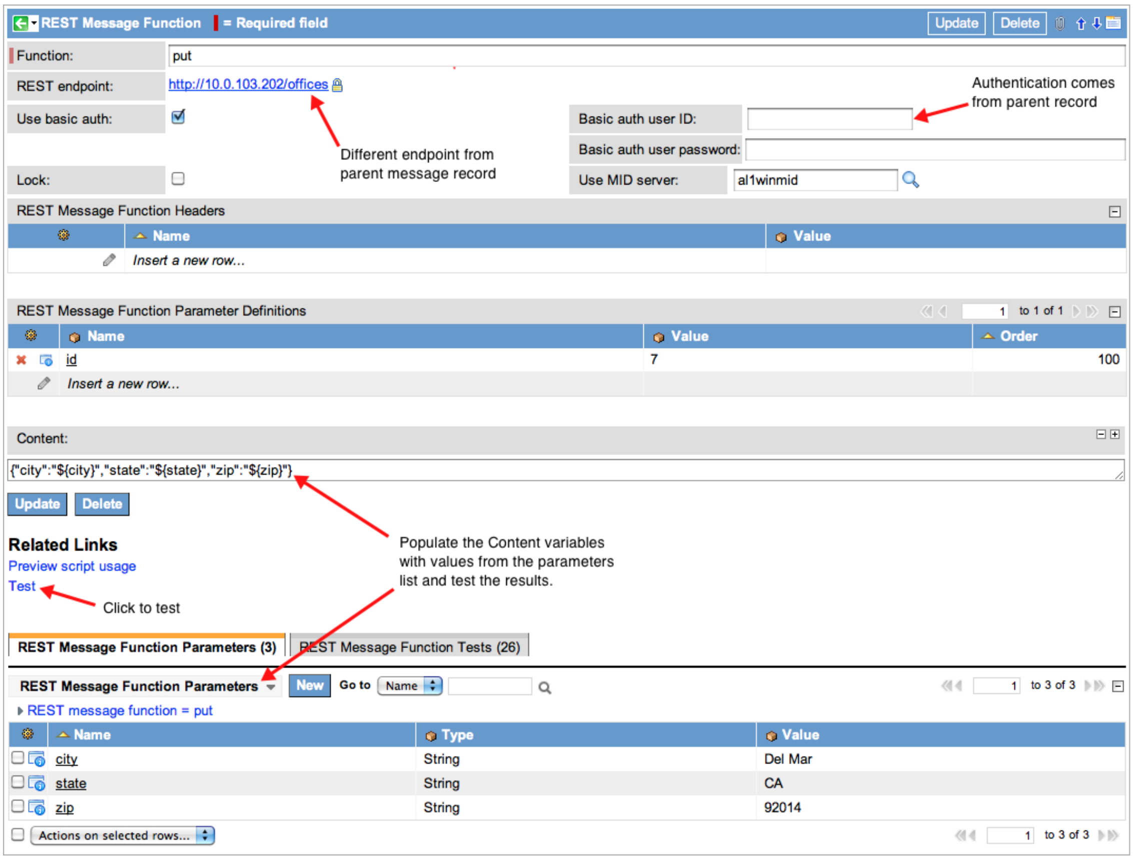
Testing a Method

To test the request, manually define the parameter values to send.

1. In the **Variable Substitutions** related list, click **New**.
2. Add test values for these variables:

   - **city**: *Del Mar*
   - **state**: *CA*
   - **zip**: *92014*

   The web service used for this test expected parameters of **city**, **state**, and **zip**. Your parameters will likely differ.


REST Message Function
= Required field
Update
Delete
Authentication comes from parent record
Different endpoint from parent message record
Populate the Content variables with values from the parameters list and test the results.
Click to test

Function:	put
REST endpoint:	http://10.0.103.202/offices
Use basic auth:	☑
Basic auth user ID:	
Basic auth user password:	
Lock:	□
Use MID server:	al1winmid

REST Message Function Headers

Name	Value
Insert a new row...	

REST Message Function Parameter Definitions

Name	Value	Order
id	7	100
Insert a new row...		

Content:

```
{"city":"${city}","state":"${state}","zip":"${zip}"}
```

Update Delete

Related Links
Preview script usage
Test

REST Message Function Parameters (3) | REST Message Function Tests (26)

REST Message Function Parameters — New — Go to Name — 1 to 3 of 3

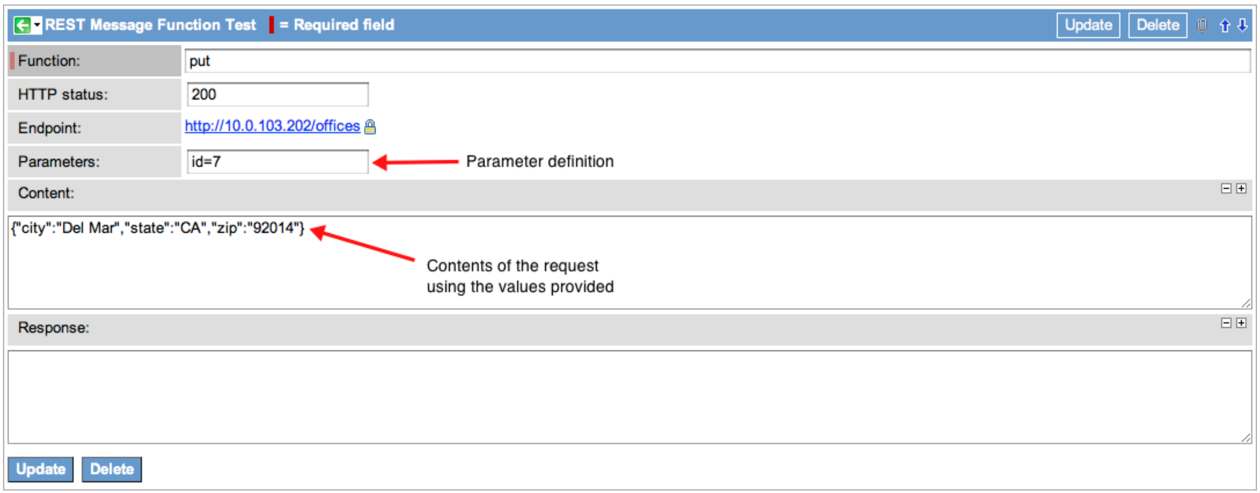
REST message function = put

Name	Type	Value
□ city	String	Del Mar
□ state	String	CA
□ zip	String	92014

□ Actions on selected rows... — 1 to 3 of 3

3. Under **Related Links**, click **Test** to determine whether the parameter definitions are configured properly.

   This action submits the record and then runs the test. The test sends the PUT request to the endpoint and updates the data at ID 7. The test results of the PUT method show that the values of **Del Mar**, **CA**, and **92014** were added to the endpoint at ID 7. In this example, the web service did not return a response body. If it had, the body would be shown in the **Response** field.



Function:	put
HTTP status:	200
Endpoint:	http://10.0.103.202/offices
Parameters:	id=7

Parameter definition

Content:

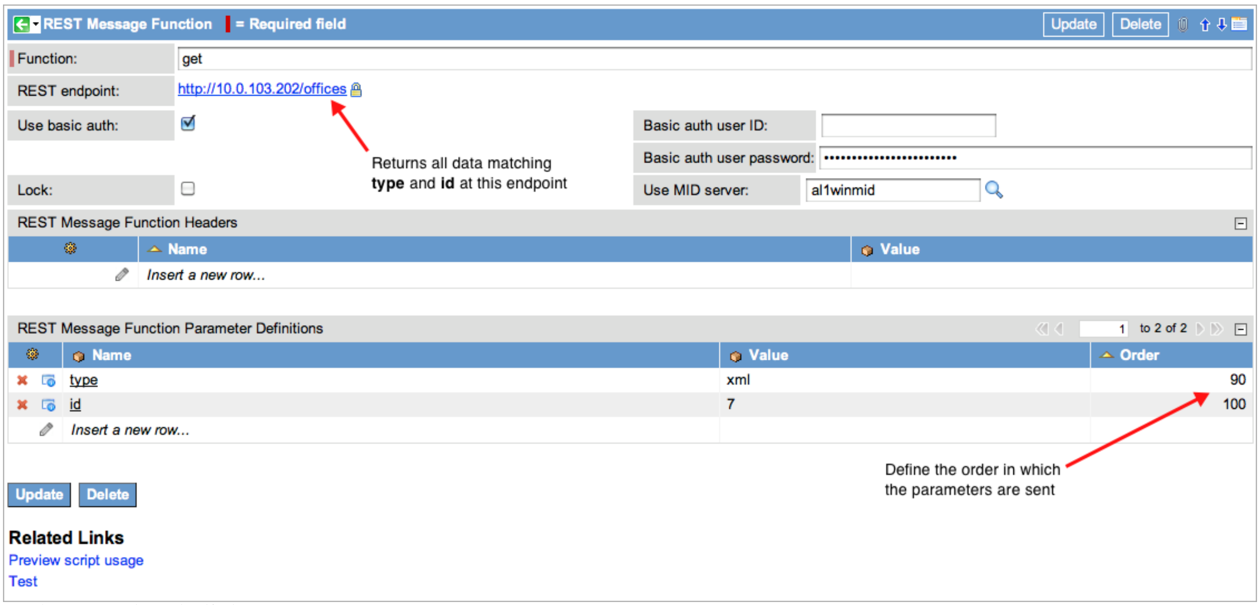
{"city":"Del Mar","state":"CA","zip":"92014"}

Contents of the request using the values provided

Response:

4. Return to the HTTP Method form for the PUT method.
5. Click the **Preview script usage** link to generate JavaScript you can use to call this method from other parts of the ServiceNow instance, such as business rules.
6. To retrieve the data just added, open the **GET** HTTP method record and specify the **Endpoint** of **/offices**.
7. Configure definitions for the parameters you want to return from the collection.

   This example includes the definition **Name** is **type** and **Value** is **xml** to return the data with ID **7** in XML format.
8. Define the **Order** in which the parameters are sent.



Function:	get
REST endpoint:	http://10.0.103.202/offices
Use basic auth:	☑
Lock:	□
Basic auth user ID:	
Basic auth user password:	••••••••••••••••••••••••
Use MID server:	al1winmid

Returns all data matching **type** and **id** at this endpoint

REST Message Function Headers

Name	Value
Insert a new row...	

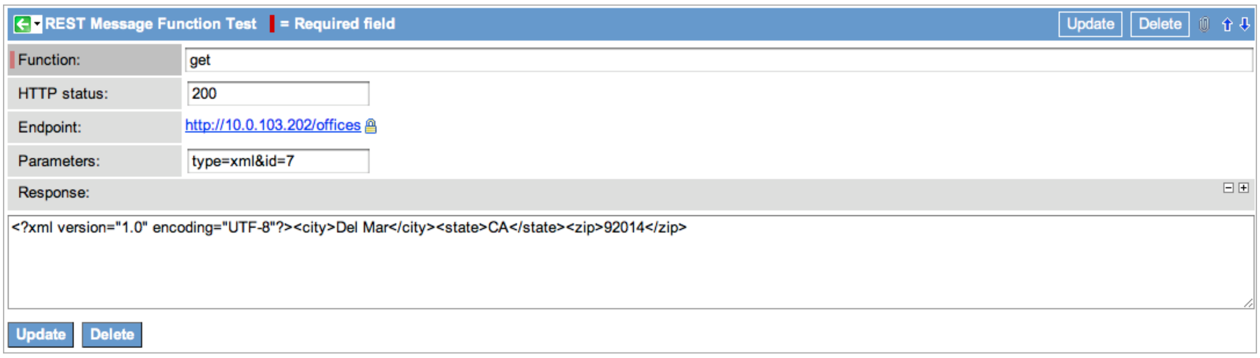
REST Message Function Parameter Definitions (1 to 2 of 2)

Name	Value	Order
type	xml	90
id	7	100
Insert a new row...		

Define the order in which the parameters are sent

9. Save the record and click **Test**.

   The response contains the XML data from the collection and all the data from ID 7.



Function:	get
HTTP status:	200
Endpoint:	http://10.0.103.202/offices
Parameters:	type=xml&id=7

Response:

```
<?xml version="1.0" encoding="UTF-8"?><city>Del Mar</city><state>CA</state><zip>92014</zip>
```

10. To generate a script template from this configuration, return to the GET record HTTP Method form and click the **Preview script usage** link.

Using Special Characters in URIs

A REST endpoint URI or variable may use special characters, such as pipe (|) characters. When using these characters in a REST message, use URL encoding [4] to escape these characters. For example, to use a parameter value of **user|title**, enter **user%7Ctitle**. Entering special characters directly may cause the REST message to fail and display the response **Invalid uri <*URI*>: Invalid query**.

Variable Substitution

Variable substitution is allowed in the URL of the endpoint, the value fields of the HTTP headers and parameters, and the **Content** field, depending on the method. The syntax for variables is `${variable_name}`. The REST message substitutes this variable with the parameter values provided when the method runs. For example, if the endpoint is expressed as **http://myserver.mycompany.com/offices/${id}**, a parameter named **id** must exist and contain a value that can be used when the method runs.

Variable substitution is used as follows for REST methods:

- **GET**: has HTTP parameter definitions that become name-value pairs in the URL when the request is sent. The sequence defined in the parameter **Order** field determines the order that the parameters are sent.
- **PUT**: uses a content expression for *putting* text content via HTTP at the endpoint site specified. Variable substitution is allowed in the **Content** field.
- **POST**: uses a content expression for *posting* text content via HTTP. Variable substitution is allowed in the **Content** field.
- **DELETE**: does not use content expressions. Variable substitution is allowed in headers and the endpoint URL.

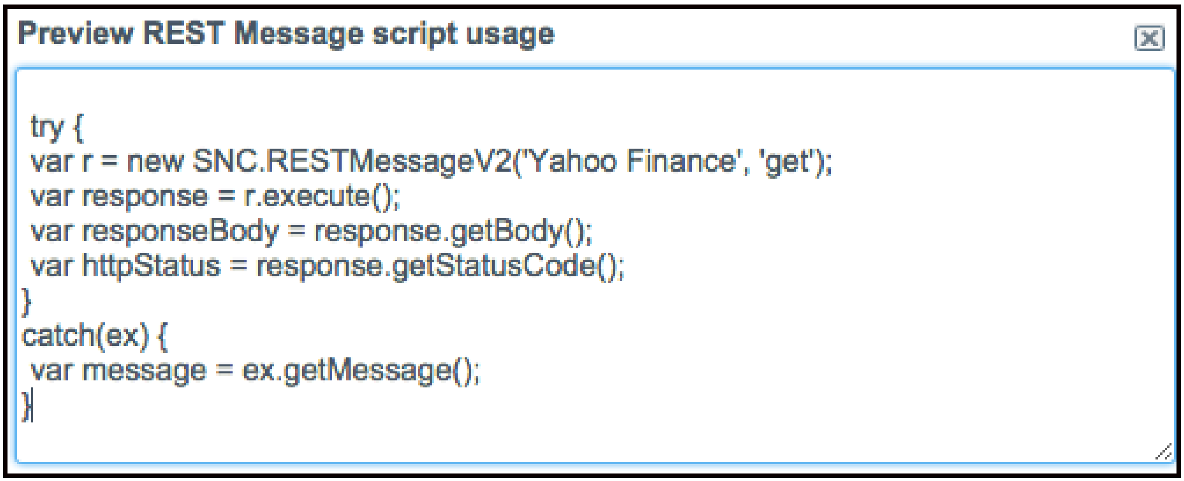
Scripting with REST

You can call REST methods from any place in the ServiceNow platform where scripting is allowed. For example, you can return data from a REST endpoint using a business rule when an event is triggered. Create a script from scratch or let the REST message preview feature create the script based on content and parameters you provide in the method record.

To generate JavaScript code to send your REST message:

1. Navigate to **System Web Services > Outbound > REST Message**.
2. Open a REST Message record.
3. In the **HTTP Methods** related list, open a method record and define the method, including any variables.
4. Save the record.
5. In the **Variable Substitutions for Tests** related list, assign a value to each variable.
6. Under **Related Links**, click **Preview script usage**.

   The instance displays the script that the REST message generated for this method.



```
 try {
 var r = new SNC.RESTMessageV2('Yahoo Finance', 'get');
 var response = r.execute();
 var responseBody = response.getBody();
 var httpStatus = response.getStatusCode();
}
catch(ex) {
 var message = ex.getMessage();
}
```

7. Copy this script and modify it as needed to use elsewhere in the instance.

Additional Scripting Methods

An API detailing scriptable methods for outbound REST messages and the returned REST response is available starting with the Fuji release. Outbound REST scripting examples are available. For older versions, see the previous version information.

Mutual Authentication

Mutual authentication causes the web service provider and consumer to authenticate with each other before communicating. ServiceNow supports mutual authentication for outbound web services, starting with the Fuji release. Mutual authentication is not available for inbound web services or for outbound web services that use a MID Server.

Enhancements

Fuji

- The glide.outbound.sslv3.disabled system property can disable the SSLv3 protocol for outbound connections.
- The labels and field help for fields on the REST Message form and related forms have been updated to use industry-standard terminology.
- Mutual authentication is available for outbound web services that do not use a MID Server.
- The RESTMessageV2 API and RESTResponseV2 API provide enhanced interfaces for sending outbound REST messages using scripts.

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/outbound-rest/concept/c_OutboundRESTWebService.html

[2] http://en.wikipedia.org/wiki/Representational_state_transfer#RESTful_web_services

[3] http://en.wikipedia.org/wiki/List_of_HTTP_header_fields

[4] https://en.wikipedia.org/wiki/Percent-encoding

RESTMessageV2 API

This article applies to Fuji. For more current information, see RESTMessage V2 [1] **at** https://developer.servicenow.com'''

The ServiceNow Wiki is no longer being updated. Please refer to the Developer Portal [1] **for the latest information.**

Overview

The RESTMessageV2 API allows you to send outbound REST messages using JavaScript. Use the RESTResponseV2 API to manage the response returned by the REST provider. For examples demonstrating how to use this API, see Scripting Outbound REST.

Note: *The outbound REST web service API is significantly enhanced with the Fuji release. This page describes current behavior. For previous version information, see Scripting with REST Functions.*

NEW: You can also find this class documented in the easy-to-read API reference on the Developer Program web site: RESTMessageV2 [2].

Method Summary

Return Type	Method
Not Applicable	RESTMessageV2() The message constructor.
Not Applicable	RESTMessageV2(String name, String methodName) The message constructor.
RESTResponse	execute() Send the REST message to the endpoint.
RESTResponse	executeAsync() Send the REST message to the endpoint asynchronously.
void	setHttpMethod(String method) Set the HTTP method this REST message performs.
void	setHttpTimeout(Number timeoutMs) Set the amount of time the REST message waits for a response from the REST provider.
void	setBasicAuth(String userName, String userPass) Set basic authentication headers for the REST message.
void	setMutualAuth(String profileName) Set the mutual authentication protocol profile for the REST message.
void	setEccCorrelator(String correlator) Associate outbound requests and the resulting response record in the ECC queue.
void	setEccParameter(String name, String value) Override a value from the database by writing to the REST message payload.
void	setMIDServer(String midServer) Configure the REST message to communicate through a MID Server.

void	setEndpoint(String endpoint)	Set the endpoint for the REST message.
void	setRequestBody(String body)	Set the body content of a PUT or POST request.
void	setRequestHeader(String name, String value)	Set an HTTP header to the specified value.
void	setStringParameter(String name, String value)	Set a REST message function variable to the specified value.
void	setStringParameterNoEscape(String name, String value)	Set a REST message function variable to the specified value without escaping XML reserved characters.
void	setQueryParameter(String name, String value)	Append a name-value parameter to the request URL.
String	getRequestBody()	Get the content of the REST message body.
String	getEndpoint()	Get the URL of the endpoint for the REST message.
String	getRequestHeader(String headerName)	Get the value for an HTTP header specified by the REST client.
Object	getRequestHeaders()	Get name and value for all HTTP headers specified by the REST client.

Constructors

RESTMessageV2()

Instantiates an empty RESTMessageV2 object. When using an object instantiated this way, you must manually specify an HTTP method and endpoint.

Parameters

- `None`

RESTMessageV2(String name, String methodName)

Instantiates a RESTMessageV2 object using information from a REST message record. You must have a REST message record defined.

Parameters:

- `name` - (String) the name of the REST message record.
- `methodName` - (String) the name of the HTTP method to use, such as GET or PUT.

Method Details

execute()

Send the REST message to the endpoint.

Parameters:

- `None`

Returns:

`RESTResponse` - the response returned by the REST provider. See RESTResponseV2 API for more information.

executeAsync()

Send the REST message to the endpoint asynchronously. The instance does not wait for a response from the web service provider when making asynchronous calls.

Parameters:

- `None`

Returns:

`RESTResponse` - the response returned by the REST provider. See RESTResponseV2 API for more information.

setHttpMethod(String method)

The HTTP method this REST message performs, such as GET or PUT. You must set an HTTP method when using the RESTMessageV2() constructor with no parameters.

Parameters:

- `method` - (String) the HTTP method to perform.

Returns:

`void`

setHttpTimeout(Number timeoutMs)

Set the amount of time the REST message waits for a response from the web service provider before the request times out.

Parameters:

- `timeoutMs` - (Number) the length of time, in milliseconds, before the call to the REST provider times out.

Returns:

`void`

setBasicAuth(String userName, String userPass)

Sets basic authentication headers for the REST message. Setting security values using this method overrides basic authentication values defined for the REST message record.

Parameters:

- `userName` - (String) the username you want to use to authenticate the REST message.
- `userPass` - (String) the password for the specified user.

Returns:

`void`

setMutualAuth(String profileName)

Set the mutual authentication protocol profile for the REST message. Setting a protocol profile using this method overrides the protocol profile selected for the REST message record.

Parameters:

- `profileName` - (String) the **Name** of the protocol profile to use for mutual authentication.

Returns:

`void`

setEccCorrelator(String correlator)

Associate outbound requests and the resulting response record in the ECC queue. This method only applies to REST messages sent through a MID Server. The correlator provided populates the **Agent correlator** field on the ECC queue record for the response. Provide a unique correlator for each outbound request to associate the correct results in the ECC queue with the request when designing asynchronous automation through a MID Server.

Parameters:

- `correlator` - (String) a unique identifier.

Returns:

`void`

setEccParameter(String name, String value)

Override a value from the database by writing to the REST message payload. This method only applies to REST messages sent through a MID Server. Use this method when a value from the REST message in the database is invalid, such as when the endpoint URL is longer than the maximum **REST endpoint** field length.

You can set only the endpoint URL using this method by passing **source** as the `name` parameter.

Parameters:

- `name` - (String) the name of the parameter, such as **source**.
- `value` - (String) the value to assign to the specified parameter.

Returns:

`void`

setMIDServer(String midServer)

Configure the REST message to communicate through a MID Server.

Parameters:

- `midServer` - (String) the name of the MID Server to use. Your instance must have an active MID Server with the specified name.

Returns:

```
void
```

setEndpoint(String endpoint)

Set the endpoint for the REST message. By default, the REST message uses the endpoint specified in the REST message record. Use this method to override this default. You must call this method when using the RESTMessageV2() constructor with no parameters.

Parameters:

- `endpoint` - (String) the URL of the REST provider you want to interface with.

Returns:

```
void
```

setRequestBody(String body)

Set the body content to send to the web service provider when using PUT or POST HTTP methods. When you set the body content using this method, variables in the body are not substituted for parameters from the REST message function record. You must explicitly define all values within the REST message body.

Parameters:

- `body` - (String) the request body to send.

Returns:

```
void
```

setRequestHeader(String name, String value)

Set an HTTP header in the REST message to the specified value.

Parameters:

- `name` - (String) the name of the header.
- `value` - (String) the value to assign to the specified header.

Returns:

```
void
```

setStringParameter(String name, String value)

Set a REST message function variable with the specified name from the REST message record to the specified value. XML reserved characters in the value are converted to the equivalent escaped characters.

Parameters:

- `name` - (String) the name of the REST message variable.
- `value` - (String) the value to assign the variable.

Returns:

`void`

setStringParameterNoEscape(String name, String value)

Set a REST message function variable with the specified name from the REST message record to the specified value. This method is equivalent to setStringParameter but does not escape XML reserved characters.

Parameters:

- `name` - (String) the name of the REST message variable.
- `value` - (String) the value to assign the variable.

Returns:

`void`

setQueryParameter(String name, String value)

Append a parameter to the end of the request URL with the form *name=value*. For example, the code `setQueryParameter("sysparm_query","active=true^ORDERBYnumber^ORDERBYDESCcategory");` appends the text *sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory* to the request URL.

Parameters:

- `name` - (String) the name of the URL parameter to pass.
- `value` - (String) the value to assign the URL parameter.

Returns:

`void`

getRequestBody()

Get the content of the REST message body.

Parameters:

- `None`

Returns:

`String` - the REST message body.

getEndpoint()

Get the URL of the endpoint for the REST message.

Parameters:

- `None`

Returns:

`String` - the URL of the REST web service provider.

getRequestHeader(String headerName)

Get the value for an HTTP header specified in the REST message. By default, this method cannot return the value for a header set automatically by the system. To grant this method access to all headers, set the property `glide.http.log_debug` to **true**.

Parameters:

- `headerName` - (String) the request header you want to get the value for.

Returns:

`String` - the value of the specified header.

getRequestHeaders()

Get HTTP headers that were set by the REST client and the associated values. This method does not return headers set automatically by the system. To configure this method to return all headers, set the property `glide.http.log_debug` to **true**.

Parameters:

- `None`

Returns:

`Object` - an Object that maps the name of each header to the associated value.

References

[1] https://developer.servicenow.com/app.do#!/api_doc?v=jakarta&id=c_RESTMessageV2API

[2] http://developer.servicenow.com/app.do#!/api_doc?to=class__restmessagev2

RESTResponseV2 API

Overview

The RESTResponseV2 API allows you to use the data returned by an outbound REST message in JavaScript code. A RESTResponseV2 object is returned by the RESTMessageV2 functions execute() and executeAsync().

Note: *The outbound REST web service API is significantly enhanced with the Fuji release. This page describes current behavior. For previous version information, see scripting with REST functions.*

NEW: You can also find this class documented in the easy-to-read API reference on the Developer Program web site: RESTResponseV2 [1].

Method Summary

Return Type	Method
void	waitForResponse(Number timeoutSecs) Set the amount of time the instance waits for the response.
Number	getStatusCode() Get the numeric HTTP status code returned by the REST provider.
String	getHeader(String name) Get the value for a specified header.
Object	getHeaders() Get all headers returned in the REST response and the associated values.
String	getBody() Get the content of the REST response body.
boolean	haveError() Indicate if there was an error during the REST transaction.
Number	getErrorCode() Get the numeric error code, if there was an error during the REST transaction.
String	getErrorMessage() Get the error message if there was an error during the REST transaction.
String	getQueryString() Get the query used for this request.

Method Detail

waitForResponse(Number timeoutSecs)

Set the amount of time the instance waits for a response from the web service provider. This method overrides the property `glide.rest.outbound.ecc_response.timeout` for this REST response.

Parameters:

- `timeoutSecs` - (Number) the amount of time, in seconds, to wait for this response.

Returns:

`void`

getStatusCode()

Get the numeric HTTP status code returned by the REST provider.

Parameters:

- `None`

Returns:

`Number` - the numeric status code returned by the REST provider, such as 200 for a successful response.

getHeader(String name)

Get the value for a specified header.

Parameters:

- `name` - (String) the name of the header that you want the value for, such as Set-Cookie.

Returns:

`String` - the value of the specified header.

getHeaders()

Get all headers returned in the REST response and the associated values.

Parameters:

- `None`

Returns:

`Object` - an Object that maps the name of each header to the associated value.

getBody()

Get the content of the REST response body.

Parameters:

- `None`

Returns:

`String` - the REST response body.

haveError()

Indicate if there was an error during the REST transaction.

Parameters:

- `None`

Returns:

`boolean` - true if there was an error, false if there was no error.

getErrorCode()

Get the numeric error code if there was an error during the REST transaction. This error code is specific to the ServiceNow platform, it is not an HTTP error code. Provide this error code if you require assistance from ServiceNow Customer Support.

Parameters:

- `None`

Returns:

`Number` - the numeric error code, such as 1 for a socket timeout.

getErrorMessage()

Get the error message if there was an error during the REST transaction.

Parameters:

- `None`

Returns:

`String` - the error message.

getQueryString()

Get the fully-resolved query sent to the REST endpoint. This query contains the endpoint URL as well as any values assigned to variables in the REST message.

Use this method only with responses to direct requests. This method is not supported for requests sent asynchronously, or requests sent using a MID server.

Parameters:

- `None`

Returns:

`String` - the fully-resolved query.

References

[1] http://developer.servicenow.com/app.do#!/api_doc?to=class__restresponsev2

Article Sources and Contributors

REST API *Source*: http://wiki.servicenow.com/index.php?oldid=251121 *Contributors*: Anat.kerry, Emily.partridge, Fuji.publishing.user, Joe.zucker, John.ramos, Joseph.messerschmidt, Vaughn.romero

Table API *Source*: http://wiki.servicenow.com/index.php?oldid=251148 *Contributors*: Fuji.publishing.user, Joe.zucker, John.ramos, Joseph.messerschmidt, Phillip.salzman

Import Set API *Source*: http://wiki.servicenow.com/index.php?oldid=251120 *Contributors*: Joe.zucker, John.ramos, Joseph.messerschmidt, Rachel.sienko

Aggregate API *Source*: http://wiki.servicenow.com/index.php?oldid=251135 *Contributors*: Fuji.publishing.user, John.ramos, Joseph.messerschmidt

Aggregate API Examples *Source*: http://wiki.servicenow.com/index.php?oldid=251136 *Contributors*: Fuji.publishing.user, John.ramos, Julie.phaviseth

Outbound REST Web Service *Source*: http://wiki.servicenow.com/index.php?oldid=250347 *Contributors*: Fuji.publishing.user, John.ramos, Joseph.messerschmidt, Mary.stromberg

RESTMessageV2 API *Source*: http://wiki.servicenow.com/index.php?oldid=251139 *Contributors*: Amy.bowman, Fuji.publishing.user, John.ramos, Joseph.messerschmidt

RESTResponseV2 API *Source*: http://wiki.servicenow.com/index.php?oldid=248346 *Contributors*: Amy.bowman, Fuji.publishing.user, Joseph.messerschmidt

Image Sources, Licenses and Contributors