

Web Services

Integrating ServiceNow via the Web

Introduction to Web Services

Web Services



Note: *This article applies to Fuji. For more current information, see [Web Services](http://docs.servicenow.com) ^[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* **The ServiceNow Wiki is no longer being updated.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

HTTP-based web services ^[2] allow diverse applications to talk to each other. ServiceNow supports both inbound (provider) and outbound (consumer) web services.

Video Tutorial

The following video tutorial explains, with use case examples, how web services work in ServiceNow instances. Provides a brief overview of the System Web Services application.

Web Services: Introduction

Inbound Web Services

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

- Direct Web Services: query tables and records directly using SOAP, REST, or other web service formats.
- ODBC Driver: report on ServiceNow data using an ODBC client, such as Microsoft Excel.
- Import Set: access the import set tables and import data through a web service interface.
- Scripted Web Services: define custom web service endpoints using JavaScript.

Outbound Web Services

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

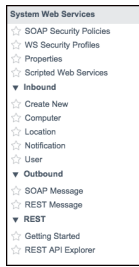
- Outbound REST
- Outbound SOAP

Roles

Role name	Description
web_service_admin	Can access the System Web Services application menu, create scripted web services, and create new import web services.

Menus and Modules

To access web service features, use these modules under the **System Web Services** application menu.



- **SOAP Security Policies:** View, create, and edit security policies for inbound SOAP web services.
- **WS Security Profiles:** View, create, and edit WS-Security profiles for inbound SOAP web services.
- **Properties:** Set properties to control the behavior of web services.
- **Scripted Web Services:** View, create, and edit scripted web services.
- **Inbound**
 - **Create New:** Create new web services for importing data.
 - **Computer:** View and edit the base system Computer [imp_computer] web service import set staging table.
 - **Location:** View and edit the base system Location [imp_location] web service import set staging table.
 - **Notification:** View and edit the base system Notification [imp_notification] web service import set staging table.
 - **User:** View and edit the base system User [imp_user] web service import set staging table.
- **Outbound**
 - **SOAP Message:** View, create, and edit outbound SOAP messages.
 - **REST Message:** View, create, and edit outbound REST messages.
- **REST**
 - **Getting Started:** View the inbound REST web service getting started guide.
 - **REST API Explorer:** Use the inbound REST web service explorer.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/web-services/reference/r_AvailableWebServices.html
- [2] https://docs.servicenow.com/bundle/helsinki-servicenow-platform/page/integrate/web-services/reference/r_AvailableWebServices.html

SOAP Web Service

SOAP



Note: This article applies to Fuji. For more current information, see *SOAP 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

The SOAP Web Services provided by ServiceNow are WS-I compliant, as outlined in the WS-I Basic Profile 1.0 ^[2]. Inbound SOAP web services comply with SOAP1.1 standards.

Concepts and Terminology

- Provider - publishes web service for clients to invoke (consume)
- Consumer - invokes / consumes published web service
- Standards
 - WSDL
 - Web Service Description Language
 - XML document describing functions, arguments, data schema, and endpoint (where / how to invoke the service, URL)
 - WSDL only necessary when generating SOAP envelope programmatically
 - SOAP
 - Simple Object Access Protocol
 - XML document usually HTTP posted to web service endpoint described in WSDL
 - SOAP:Envelope / SOAP:Header / SOAP:Body
 - HTTP
 - Hyper-Text Transfer Protocol
 - POST vs GET - Web Service is POSTed
 - XML vs. Form POST - Web Service is XML

Web Service Provider

ServiceNow publishes its underlying table structures and associated data with the following Web Service methods:

- Direct Web Services: Use a URL query to request a ServiceNow table's WSDL.
- Web Service Import Sets: Use import tables and transform maps to automate Web Service requests to ServiceNow tables.
- Scripted Web Services: Use custom Javascript to execute Web Services requests.



Note: SOAP messages are sent with the assumption that the recipient is XML compliant. No encoding is applied to the SOAP message.

Web Service Consumer

Consuming external web services is achieved using Javascript objects that represent the web service SOAP ^[3] envelope and the subsequent SOAP HTTP request that submits the request. Web Service Consumer documents these programmatic constructs as well as examples of how to invoke web services.

WSDL

All ServiceNow tables and import sets dynamically generate WSDL XML documents that describe its table schema and available operations. You can get a WSDL format by issuing a URL targeting a ServiceNow table with the **WSDL** parameter, for example:

```
https://myinstance.service-now.com/incident.do?WSDL
```

All dynamically generated and served ServiceNow WSDL accessible via HTTP is available for use under the terms defined in the Open Source Initiative OSI - Apache License, Version 2.0 license agreement.

[*OSI - Apache License, Version 2.0*](#)

Apache License, Version 2.0

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such

Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may

choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

Direct Web Services

A SOAP direct web service is available for any table in the system provided the correct access control is setup. The supported format of the incoming message is document style literal XML SOAP documents (Document/Literal). To retrieve the direct web service WSDL ^[4] description and XML schema, point to the relative URL of **<tablename>.do?WSDL**. For example, to retrieve the WSDL for the Incident table on the online demo system, use the following URL:

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

Web Service Import Sets

Web Service Import Sets provide a web service interface to import set tables. This type of web service will transform the incoming data synchronously based on the associated transform maps by default.

SOAP Roles

To use SOAP web services, you must have the appropriate role for the operation you want to perform. Additionally, you must have any other roles required to access the target tables.

Role	Description
soap	Can perform all SOAP operations.
soap_create	Can insert new records.
soap_delete	Can delete existing records.
soap_ecc	Can query, insert, and delete records on the Queues [ecc_queue] table.
soap_query	Can query record information.
soap_query_update	Can query record information and update records.
soap_script	Can run scripts that specify a .do endpoint. This role is required for running Scripted Web Services.
soap_update	Can update records.

Overriding the SOAP Endpoint

The SOAP End-point address (where the SOAP message is posted) is consistent with the endpoint of the WSDL. In some cases, however, the WSDL may reference an incorrect endpoint URL. If this happens, it is necessary to over-ride the generated URL by creating the property **com.glide.soap_address_base_url** to contain the new URL. You may have to add the property manually as this is not an out-of-box property.

For instance, a generated SOAP Endpoint may look like this:

```
https://instance.service-now.com/incident.do?SOAP
```

You can specify a property to define the endpoint such that it goes through a proxy by setting the property:

```
com.glide.soap_address_base_url = "https://myproxy.mycompany.com/service-now/"
```

This will result in the endpoint being generated to appear as:

```
https://myproxy.mycompany.com/service-now/incident.do?SOAP
```

Enabling HTTP Compression

By default, the SOAP request is accepted un-compressed and the result of the request is returned un-compressed. To enable HTTP compression using [gzip^[5]] when sending in your SOAP request, set the following HTTP header:

```
Content-Encoding: gzip
```

To receive the SOAP response compressed using [gzip^[5]] send in your SOAP request with the following HTTP header:

```
Accept-Encoding: gzip
```

Debugging Incoming SOAP Envelope

To capture incoming SOAP envelope XML in the system log, add the property `glide.processor.debug.SOAPProcessor` with a value of **true**.

When enabled, this property adds the incoming SOAP envelope in the **Message** field of the system log. Disable this debugging feature as soon as you are finished so that the log is not overwhelmed with excessive and unnecessary debugging information.

Preventing Empty Elements in SOAP Messages

By default, ServiceNow does not omit empty elements, elements with NULL or NIL values, from SOAP messages.

To prevent SOAP responses from containing empty elements, an administrator can create a system property called **glide.soap.omit_null_values** and set the value to **true**. This behavior is compliant with the WSDL as all elements in a SOAP message have a **minOccurs=0** attribute and are therefore optional. In addition, this behavior prevents the instance from creating inefficient SOAP messages containing a large number of empty elements.

Set this property to **false** to allow SOAP messages to search for existing fields with empty values. For example, if an administrator wants to search for incidents with an empty **Assigned to** field from a SOAP message, then the SOAP message must be able to send an empty value for this field.



Note: Changing the value of this property may cause unintended actions in existing web service integrations. Administrators are strongly encouraged to carefully test the new behavior to ensure that existing integrations process empty elements as expected.

Enhancements

Fuji

- The property `glide.soap.request_processing_timeout` has a default value of 60 seconds.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-soap/concept/c_SOAPWebService.html
- [2] <http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html>
- [3] <http://www.w3.org/TR/soap/>
- [4] <http://www.w3.org/TR/wsdl>
- [5] <http://en.wikipedia.org/wiki/Gzip>

Direct Web Services



Note: This article applies to Fuji. For more current information, see *Direct Web Services* ^[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

A direct web service is available for any table in the system if you configure the correct access control. The supported format of the incoming message is document style literal XML SOAP documents (Document/Literal wrapped ^[2]). To retrieve the direct web service WSDL ^[4] description and XML schema for any given table, point to the relative URL of `<tablename>?WSDL`. For example, to retrieve the WSDL for the Incident table on the online demo system, use the following URL:

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

Notes:

- When using Direct Web Services, input fields should not be left empty. If they are not required, they should be omitted entirely.
- Default return for any query is limited to 250 records.

Available APIs

ServiceNow supports multiple APIs for direct web services.

- SOAP - <https://en.wikipedia.org/wiki/SOAP>
 - REST - https://en.wikipedia.org/wiki/REST#Applied_to_web_services
 - CSV - https://en.wikipedia.org/wiki/Comma-separated_values
 - EXCEL - https://en.wikipedia.org/wiki/Microsoft_Excel_file_format#File_formats
 - JSONv2 - <http://json.org/>
 - PDF - https://en.wikipedia.org/wiki/Portable_Document_Format
 - RSS - <https://en.wikipedia.org/wiki/RSS>
 - XML - <https://en.wikipedia.org/wiki/XML>
 - ODBC - https://en.wikipedia.org/wiki/Open_Database_Connectivity
-

Using Form Views to Limit/Extend Query Response

On occasion, there is a need to limit the number of field values being returned from a SOAP query (get or getRecords) invocation. Specifying a form view has the effects of:

1. limiting the response elements to contain only the fields on the view
2. specifying reference record field values from referenced fields eg. caller_id.email, this will cause the value of the caller's email to be returned in the SOAP response

To enable form views for SOAP queries, you may configure the property **com.glide.soap.view** to be the name of the view you wish to use for all SOAP query response (the default is **soap_response**). You may also specify the view name as a URL parameter **sysparm_view=<view name>** when making the SOAP call, for example:

```
https://<instance name>.service-now.com/incident.do?SOAP&sysparm_view=ess
```

If a specified view name does not exist, the default behavior is to respond with all fields.

Extended Query Parameters

The following parameters, when specified as elements of input parameters to SOAP query functions such as get, getKeys, and getRecords, has the additional behavior of filtering and modifying the results that are returned.



Note: Extended query element names are preceded by two underscore characters.

Extended Query Parameter	Description	Example
__encoded_query	Specify an encoded query string to be used in filtering the returned results. The encoded query string format is similar to the value that may be specified in a sysparm_query URL parameter. You may refer to the encoded query building example in the RSS feed generator examples.	<pre><__encoded_query>active=true^category='hardware'</__encoded_query></pre> <p>The example above illustrates a query for active records that have a category of 'hardware'</p>
__order_by	Instruct the returned results to be ordered by the specified field	<pre><__order_by>priority</__order_by></pre>

__order_by_desc	Instruct the returned results to be ordered by the specified field, in descending order	<__order_by_desc>opened_date</__order_by_desc>
__exclude_columns	Specify a list of comma delimited field names to exclude from the result set	<__exclude_columns>sys_created_on,sys_created_by,caller_id,priority</__exclude_columns>
__limit	Limit the number of records that are returned	<__limit>100</__limit>
__first_row	Instruct the results to be offset by this number of records from the beginning of the set. When used with __last_row has the effect of querying for a window of results. The results are inclusive of the first row number.	<__first_row>250</__first_row>

__last_row	Instruct the results to be limited by this number of records from the beginning of the set, or the __start_row value when specified. When used with __first_row has the effect of querying for a window of results. The results are less than the last row number, and does not include the last row.	<__last_row>500</__last_row>
__use_view	Specify a Form view by name, to be used for limiting and expanding the results returned. When the form view contains deep referenced fields eg. <i>caller_id.email</i> , this field will be returned in the result as well	<__use_view>soap_view</__use_view>

Perl Example - querying all incidents 5 records at a time

This example query queries all incidents, orders by the *number* field, and retrieves the first 5 records. You can easily extend this example to retrieve a set of predefined records, *n* number of records each query, simulating a *windowed* querying client. Using a *windowed* query mechanisms overcomes the default limitation of only getting a maximum of 250 records per query.

```
#!/usr/bin/perl -w

#use SOAP::Lite ( +trace => all, matype => {} );
use SOAP::Lite;
```

```

# basic auth using the ITIL user
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
    return 'itil' => 'itil';
}

# specify the endpoint to connect
my $soap = SOAP::Lite
    -> proxy('https://<instance name>.service-now.com/incident.do?SOAP');

my $method = SOAP::Data->name('getRecords')
    ->attr({xmlns => 'http://www.service-now.com/'});

# get all incidents with a window of 5, starting at row 0, and less
than row 5 (total of 5 records)
my @params = ( SOAP::Data->name(__first_row => '0') );
push(@params, SOAP::Data->name(__last_row => '5') );
# the last row number can also be taken as the 'limit' offset by the
starting first row

# order by the number field
push(@params, SOAP::Data->name(__order_by => 'number') );

my $result = $soap->call($method => @params);
print_fault($result);
print_results($result);

sub print_results {
    my ($result) = @_;

    my %keyHash = %{ $result->body->{'getRecordsResponse'} };

    my $i = 0;
    my $size = @{$keyHash{'getRecordsResult'}};
    for ($i=0; $i<$size; $i++) {
        my %record = %{ $keyHash{'getRecordsResult'}[$i] };
        print "----- $i
-----\n";

        foreach my $kk (keys %record) {
            # print only the number of the incident
            if ($kk eq "number") {
                print "$kk=$record{$kk}\n";
            }
        }
    }
}

sub print_fault {

```

```

my ($result) = @_;

if ($result->fault) {
    print "faultcode=" . $result->fault->{'faultcode'} . "\n";
    print "faultstring=" . $result->fault->{'faultstring'} . "\n";
    print "detail=" . $result->fault->{'detail'} . "\n";
}
}

```

Return Display Value for Reference Variables

When you query a record using a *get* or *getRecords* function the instance returns all fields associated with that record. The fields are often reference fields that contain a *sys_id* for a record on another table. Use one of these options if you want the display value for the field to be returned instead of the *sys_id*:

1. Add the property `glide.soap.return_displayValue` to your system properties, and every SOAP request will return a display value for a reference field.
2. Add the parameter **displayvalue=true** to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field as a string, instead of the *sys_id*. The SOAP URL would look as follows:

`https://<instance name>.service-now.com/incident.do?displayvalue=true&SOAP`

3. Add the parameter **displayvalue=all** to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field, in addition to the *sys_id*. The response element name for the display value field will be prefixed with *dv_* such as *dv_caller_id*.

Retrieving Journal Entries

To get the contents of a journal field, make a second soap request against the **sys_journal_field** table to pull the appropriate journal records back for the record in question.

The URL for the WSDL would be in the following format

```
https://instance-name.service-now.com/sys_journal_field.do?WSDL
```

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

```

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

```


Retrieving Choice Fields

To retrieve or set choice fields, use the choice **Value** not the **Label**. For example, if you want to retrieve a list of all closed incidents use the numerical value for **Closed**, which is 7 by default.

```
<state>7</state>
```

To see a list of choice values:

1. Navigate to the form containing the choice field. For example, navigate to **Incident > Open** and select an incident.
2. Right-click the choice value field and select **Configure Dictionary** (**Personalize Dictionary** in versions prior to Fuji). For example, configure the dictionary for the **State** field.
3. From the **Choices** related list, note the value for the label you want to query. For example, note that the **Closed** choice has a value of 7.

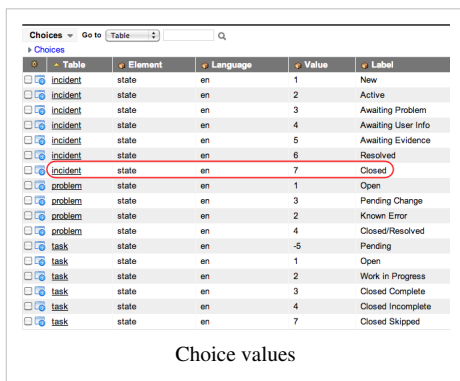


Table	Element	Language	Value	Label
incident	state	en	1	New
incident	state	en	2	Active
incident	state	en	3	Awaiting Problem
incident	state	en	4	Awaiting User Info
incident	state	en	5	Awaiting Evidence
incident	state	en	6	Resolved
incident	state	en	7	Closed
problem	state	en	1	Open
problem	state	en	3	Pending Change
problem	state	en	2	Known Error
problem	state	en	4	Closed/Resolved
task	state	en	-5	Pending
task	state	en	1	Open
task	state	en	2	Work in Progress
task	state	en	3	Closed Complete
task	state	en	4	Closed Incomplete
task	state	en	7	Closed Skipped

Choice values

Persisting HTTP session across all SOAP calls

Because ServiceNow uses stateless SOAP invocations, each SOAP call creates a new user session that persists until it expires. In circumstances when a SOAP client makes many calls in a short amount of time, you may want to re-use a single HTTP session for all SOAP calls. To create a single user session and re-use it for all inbound SOAP calls, develop your SOAP client following these guidelines:

- Use a module like HTTP::Cookies to create a "cookie jar".
- Save the cookies returned by ServiceNow after each request (handled automatically by HTTP::Cookies).
- Re-send the cookies in the cookie jar with each subsequent request.



Note: If you have enabled the session rotation high security setting, it will immediately invalidate the JSESSIONID returned from the server with the first response header. The second response includes a new JSESSIONID.

In perl, you can automatically save and send cookies with the following code:

```
use HTTP::Cookies;

#we want to store and re-send cookies
my $cookies = HTTP::Cookies->new(ignore_discard => 1);

my $soap = SOAP::Lite
    -> proxy('http://localhost:8080/glide/ecc_queue.do?SOAP');

#Set the cookie jar
$soap->transport->cookie_jar($cookies);
```

Compatibility for Clients Generated from WSDL

Specifying a Unique Namespace for each Table

The property `glide.wSDL.definition.use_unique_namespace` ensures that each table's Direct Web Service WSDL has a unique `targetNamespace` attribute. This property is true by default, which requires a table's Direct Web Service WSDL to use a `targetNamespace` value of `http://www.service-now.com/<table name>`. When false (or when the property is not present), all tables use the same `targetNamespace` value of `http://www.service-now.com`. Since all tables also share the same operation names, a Web Service client attempting to consume more than one ServiceNow Web Service would be unable to differentiate between requests between multiple tables. Using a unique `targetNamespace` value allows Web Services clients to distinguish requests between multiple tables.

For example, the Direct Web Service WSDL for the incident table uses this `targetNamespace` value.

```
<wSDL:definitions xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
  xmlns:http="http://schemas.xmlsoap.org/wSDL/http/"
  xmlns:tns="http://www.service-now.com/incident"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:mime="http://schemas.xmlsoap.org/wSDL/mime/"
  xmlns:soap="http://schemas.xmlsoap.org/wSDL/soap/"
  targetNamespace="http://www.service-now.com/incident">
<wSDL:types>
<xsd:schema elementFormDefault="unqualified"
  targetNamespace="http://www.service-now.com/incident">
```

Setting Namespace Requirements

ServiceNow's WSDL schema by default declares an attribute of `elementFormDefault="unqualified"`. This attribute indicates whether or not locally declared elements must be qualified by the target namespace in an instance document. If the value of this attribute is 'unqualified', then locally declared elements should not be qualified by the target namespace. If the value of this attribute is 'qualified', then locally declared elements must be qualified by the target namespace.

However, this is incompatible with the way clients generated from WSDL (.NET, Axis2, webMethods, ect.) process the embedded schema, it removes the schema namespace as a result, making the web service response unparseable.

To overcome this compatibility issue, a boolean property called `glide.wSDL.schema.UnqualifiedElementFormDefault` is introduced. This property has the value of **true** by default, setting it to **false** will make clients generated from WSDL able to parse the return value of the web service invocation. You can modify this property using the Web Services properties page at **System Properties > Web Services**.

This property sets the elementFormDefault attribute of the embedded XML schema to the value of unqualified, if set to true. This attribute indicates whether or not locally declared elements must be qualified by the target namespace in an instance document. If the value of this attribute is 'unqualified', then locally declared elements should not be qualified by the target namespace. If the value of this attribute is 'qualified', then locally declared elements must be qualified by the target namespace. For compatibility with Clients generated from WSDL (.NET, Axis2, webMethods, ect.) set this value to false. This value defaults to true.

For further documentation, follow this URL: http://wiki.service-now.com/index.php?title=Web_Services

Yes / No

System property for elementFormDefault attribute

Allowing Duplicate Service Names

By default, service names from dynamically generated WSDL are unique and have the following format:

```
ServiceNow_<table name>
```

To allow duplicate service names, administrators can set the `glide.wSDL.unique_service_name` property to **false**. Create the property if it does not exist.

Enhancements

Fuji

- You can control web service access to tables using the **Allow web service interaction** check box.
- The ?SOAP endpoint accepts the *elementFormDefault=qualified* request parameter. See KB0546188 ^[3] for more information.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-soap/concept/c_DirectWebServices.html
 [2] <http://www.ibm.com/developerworks/webservices/library/ws-whichwsdl/#N1021A>
 [3] https://hi.service-now.com/kb_view.do?sysparm_article=KB0546188

Direct SOAP API



Note: This article applies to Fuji and earlier releases. For more current information, see *SOAP Direct Web Service API Functions* ^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Direct SOAP API

Please note that the default return for any query is limited to 250 records.

The standard SOAP API is a set of small, globally defined functions that can be performed on a targeted resource. The targeted resource (or table) is defined in the URL by the format **`https://<instance name>.service-now.com/<table name>.do`**.

Data Retrieval API

Method Summary	Description
getKeys	Query the targeted table by example values and return a comma delimited list of sys_id .
getRecords	Query the targeted table by example values and return all matching records and their fields.
get	Query a single record from the targeted table by sys_id and return the record and its fields.
aggregate	Query using and aggregate functions SUM, COUNT MIN, MAX and AVG. To enable the aggregate functions, activate the Aggregate Web Service Plugin.

Data Modification API

Method Summary	Description
insert	Creates a new record for the table targeted in the URL.
insertMultiple	Creates multiple new records for the table targeted in the URL. To enable multiple inserts, activate the Web Service Insert Multiple Plugin .
update	Updates a existing record in the targeted table in the URL, identified by the mandatory sys_id field.
deleteRecord	Delete a record from the targeted table by supplying its sys_id .
deleteMultiple	Delete multiple records from the targeted table by example values.

Data Retrieval API

getKeys

Query the targeted table by example values and return a comma delimited list of **sys_id**.

Input Fields

Any field value in the targeted table.

Output Fields

A SOAP response element **sys_id** that contains a comma delimited list of sys_ids

Sample SOAP Messages

Sample SOAP request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getKeys>
      <category>hardware</category>
    </inc:getKeys>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getKeysResponse>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56,46e57642a9fe1981000b96a5dca501ff,46f1784ba9fe19810018aa27fbb23482</sys_id>
      <count>7</count>
    </getKeysResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

For language-specific **getKeys** examples, see:

- Perl SOAP::Lite

- Java Apache Axis2
- Microsoft .NET
- Python

getRecords

Query the targeted table by example values and return all matching records and their fields.

Input Fields

Any field value in the targeted table.

Output Fields

The **getRecordResponse** element may contain 1 or more **getRecordsResult** elements that encapsulate elements representing the field values of records matching the query.

Sample SOAP Messages

Sample SOAP request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getRecords>
      <number>INC0000002</number>
    </inc:getRecords>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP request using an encoded query to filter where incident number is INC0000001 or INC0000002

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getRecords>
      <__encoded_query>number=INC0000001^ORnumber=INC0000002</__encoded_query>
    </inc:getRecords>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response containing 1 record

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getRecordsResponse>
      <getRecordsResult>
        <caller_id>5137153cc611227c000bbd1bd8cd2007</caller_id>
        <caller_id.email>david.loo@service-now.com</caller_id.email>
        <closed_at/>
        <number>INC0000002</number>
        <opened_at>2009-12-14 23:07:12</opened_at>
        <short_description>Can't get to network file shares</short_description>
      </getRecordsResult>
    </getRecordsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

```
    </getRecordsResult>
  </getRecordsResponse>
</soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response that contains more than 1 record

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getRecordsResponse>
      <getRecordsResult>
        <caller_id>5137153cc611227c000bbd1bd8cd2006</caller_id>
        <caller_id.email>rick.berzle@yourcompany.com</caller_id.email>
        <closed_at>2009-12-17 22:55:16</closed_at>
        <number>INC0000009</number>
        <opened_at>2009-12-16 22:50:23</opened_at>
        <short_description>Reset my password</short_description>
      </getRecordsResult>
      <getRecordsResult>
        <caller_id>5137153cc611227c000bbd1bd8cd2005</caller_id>
        <caller_id.email>fred.luddy@yourcompany.com</caller_id.email>
        <closed_at>2009-12-15 22:54:55</closed_at>
        <number>INC0000010</number>
        <opened_at>2009-12-10 22:53:02</opened_at>
        <short_description>Need Oracle 10GR2 installed</short_description>
      </getRecordsResult>
    </getRecordsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

For language-specific **getRecords** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

get

Query a single record from the targeted table by **sys_id** and return the record and its fields

Input Fields

An element **<sys_id>** identifying the **sys_id** of the record to be retrieved.

Output Fields

A **getResponse** element encapsulating all field values for the record retrieved.

Sample SOAP Messages

Sample SOAP request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:get>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </inc:get>
  </soapenv:Body>
</soapenv:Envelope>

```

The resulting response of a **get** function call looks like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soap:Body>
    <getResponse xmlns="http://www.service-now.com/incident">
      <active>1</active>
      <approval>not requested</approval>
      <assigned_to>46c6f9efa9fe198101ddf5eed9adf6e7</assigned_to>
      <caller_id>46b673a6a9fe19810007ab03cbd5849d</caller_id>
      <category>network</category>
      <cmdb_ci>0c43f35dc61122750182c132a29e3243</cmdb_ci>
      <comments>Testing</comments>
      <contact_type>phone</contact_type>
      <due_date>2007-10-28 13:29:45</due_date>
      <escalation>0</escalation>
      <impact>3</impact>
      <incident_state>1</incident_state>
      <knowledge>0</knowledge>
      <location>1081761cc611227501d063fd3475a2de</location>
      <made_sla>1</made_sla>
      <notify>1</notify>
      <number>INC10055</number>
      <opened_at>2007-09-18 00:32:09</opened_at>
      <opened_by>46bac3d6a9fe1981005f299d979b8869</opened_by>
      <priority>0</priority>
    </getResponse>
  </soap:Body>
</soap:Envelope>

```

```

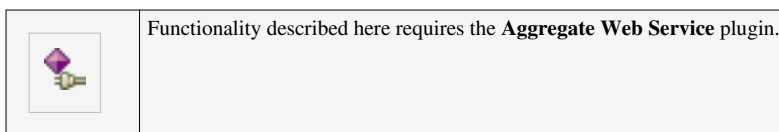
    <reassignment_count>0</reassignment_count>
    <severity>0</severity>
  </getResponse>
</soap:Body>
</soap:Envelope>

```

For language-specific **get** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

aggregate



Query a table using an aggregate function like SUM, COUNT, MIN, MAX, AVG.

Input Fields

Any element of the target table. In addition one or more of the **aggregate** functions (SUM, AVG etc...). A **group by** and a **having** clause may also be added.

Output Fields

A **aggregateResponse** element encapsulating all field values for the record retrieved.

Sample SOAP Messages

Sample SOAP request using **COUNT** aggregate function.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/"

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:m="http://www.service-now.com"
      xmlns:tns="http://www.service-now.com/map"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregate>
      <COUNT>number</COUNT>
      <active>true</active>
    </aggregate>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```


The resulting response of a **COUNT** aggregate function call looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:m="http://www.service-now.com"
    xmlns:tns="http://www.service-now.com/map"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregateResponse>
      <aggregateResult>
        <avg>2.7200</avg>
      </aggregateResult>
    </aggregateResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample SOAP request using **AVG** aggregate function with a **GROUP BY** clause.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:m="http://www.service-now.com"
    xmlns:tns="http://www.service-now.com/map"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregate xmlns="http://www.service-now.com">
      <GROUP_BY>category</GROUP_BY>
      <active>true</active>
      <AVG>severity</AVG>
    </aggregate>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The resulting response of a **AVG** aggregate function call with a **GROUP BY** clause looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
```

```

xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:m="http://www.service-now.com"
        xmlns:tns="http://www.service-now.com/map"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregateResponse>
      <aggregateResult>
        <avg>1.0000</avg>
        <category>database</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>hardware</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>inquiry</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.0000</avg>
        <category>network</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.6923</avg>
        <category>software</category>
      </aggregateResult>
    </aggregateResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Sample SOAP request using an encoded query to filter the aggregate:

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/"

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:m="http://www.service-now.com"
        xmlns:tns="http://www.service-now.com/map"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>

```

```

    <aggregate>
      <COUNT>number</COUNT>
      <active>true</active>
      <__encoded_query>number=INC0000001^ORnumber=INC0000002</__encoded_query>
    </aggregate>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Sample aggregate request using Having to narrow the results. Having takes four fields each delimited by "^": the aggregate type, the field of the aggregate, the operation type, and the value to compare . More than one Having can be added to the request so you can and Having expressions, but there is no support for OR.

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/"

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:m="http://www.service-now.com"
      xmlns:tns="http://www.service-now.com/map"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregate>
      <COUNT>sys_id</COUNT>
      <GROUP_BY>internal_type</GROUP_BY>
      <HAVING>COUNT^*^>^10</HAVING>
      <HAVING>COUNT^*^<^20</HAVING>
      <COUNT>sys_id</COUNT>
      <active>true</active>
    </aggregate>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Data Modification API

insert

Creates a new record for the table targeted in the URL

Input Fields

All fields from the targeted table, excluding system fields. Fields configured as mandatory in the System Dictionary are reflected in the WSDL with the attribute `minOccurs=1`.

Output Fields

- Regular tables: The **sys_id** field and the display value of the target table are returned.
- Import set tables: The **sys_id** of the import set row, the name of the transformed target table (**table**), the **display_name** for the transformed target table, the **display_value** of the transformed target row, and a **status**

field, which can contain **inserted**, **updated**, or **error**. There can be an optional **status_message** field or an **error_message** field value when *status=error*. When an insert did not cause a target row to be transformed, e.g. skipped because a key value is not specified, the **sys_id** field will contain the sys_id of the import set row rather than the targeted transform table.

- Import set tables with multiple transforms: The response from this type of insert will contain multiple sets of fields from the regular import set table insert wrapped in a multiInsertResponse parent element. Each set will contain a **map** field, showing which transform map created the response.

Sample SOAP Messages

The following example shows an insert that specifies the short description only:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope xmlns:tns="http://www.service-now.com/incident"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:m="http://www.service-now.com"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <insert xmlns="http://www.service-now.com">
      <short_description xsi:type="xsd:string">This is a test</short_description>
    </insert>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The resulting response looks like this:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:m="http://www.service-now.com"
  xmlns:tns="http://www.service-now.com/incident"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <insertResponse xmlns="http://www.service-now.com">
      <sys_id>6b06494fc611227d00b5f87caf618831</sys_id>
    </insertResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

For language-specific **insert** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

insertMultiple

Creates multiple new records for the table targeted in the URL.

Input Fields

The **insertMultiple** element may contain 1 or more **record** tags that contains all fields from the targeted table, excluding system fields. Limit the number of records inserted in a single operation to no more than 200. You can gradually increase this number with subsequent exports if the increase does not negatively impact instance performance.

Output Fields

The insertMultipleResponse tag followed by 1 or more record tags that contains:

- Regular tables: The **sys_id** field and the display value of the target table are returned.
- Import set tables: The **sys_id** of the import set row, the name of the transformed target table (**table**), the **display_name** for the transformed target table, the **display_value** of the transformed target row, and a **status** field, which can contain **inserted**, **updated**, or **error**. There can be an optional **status_message** field or an **error_message** field value when *status=error*. When an insert did not cause a target row to be transformed, e.g. skipped because a key value is not specified, the **sys_id** field will contain the sys_id of the import set row rather than the targeted transform table.
- Import set tables with multiple transforms: The response from this type of insert will contain multiple sets of fields from the regular import set table insert wrapped in a multiInsertResponse parent element. Each set will contain a **map** field, showing which transform map created the response.

Sample SOAP Messages - regular table

The following example shows an insert that specifies the short description only:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:insertMultiple>
      <record>
        <short_description>this is test 1</short_description>
      </record>
      <record>
        <short_description>this is test 2</short_description>
      </record>
      <record>
        <short_description>this is test 3</short_description>
      </record>
    </inc:insertMultiple>
  </soapenv:Body>
</soapenv:Envelope>
```

The resulting response looks like this:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <insertMultipleResponse>
```

```

<insertResponse>

  <sys_id>168160ad4a36231200a89091281dc803</sys_id>

  <number>INC0055180</number>

</insertResponse>

<insertResponse>

  <sys_id>1681622e4a36231200a8909115e5c388</sys_id>

  <number>INC0055181</number>

</insertResponse>

<insertResponse>

  <sys_id>1681626e4a36231200a89091fa3c0aa8</sys_id>

  <number>INC0055182</number>

</insertResponse>

</insertMultipleResponse>

</soapenv:Body>
</soapenv:Envelope>

```

Sample SOAP Messages - import set table

The following example shows an insert that specifies the short description only:

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:imp="http://www.service-now.com/imp_notification">
  <soapenv:Header/>
  <soapenv:Body>
    <imp:insertMultiple>!-->
      <imp:record>
        <imp:message>one</imp:message>
        <imp:uuid>a</imp:uuid>
      </imp:record>
      <imp:record>
        <imp:message>two</imp:message>
        <imp:uuid>b</imp:uuid>
      </imp:record>
      <imp:record>
        <imp:message>three</imp:message>
        <imp:uuid>c</imp:uuid>
      </imp:record>
    </imp:insertMultiple>
  </soapenv:Body>
</soapenv:Envelope>

```

The resulting response looks like this:

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:imp="http://www.service-now.com/imp_notification">
  <soapenv:Header/>
  <soapenv:Body>
    <insertMultipleResponse>
      <insertResponse>
        <sys_id>1296b3ab0a0a0b5b73e966fbfab7acde</sys_id>
        <table>incident</table>
        <display_name>number</display_name>
      </insertResponse>
    </insertMultipleResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

```

        <display_value>INC0010033</display_value>

        <status>ignored</status>

        <status_message>No field values changed</status_message>
    </insertResponse>
    <insertResponse>

        <sys_id>1296b48e0a0a0b5b62513bb5974a7d96</sys_id>

        <table>incident</table>

        <display_name>number</display_name>

        <display_value>INC0010034</display_value>

        <status>ignored</status>

        <status_message>No field values changed</status_message>
    </insertResponse>
    <insertResponse>

        <sys_id>1296b58b0a0a0b5b468f534659538b9a</sys_id>

        <table>incident</table>

        <display_name>number</display_name>

        <display_value>INC0010035</display_value>

        <status>ignored</status>

        <status_message>No field values changed</status_message>
    </insertResponse>
</insertMultipleResponse>
</soapenv:Body>
</soapenv:Envelope>

```

update

Updates an existing record in the targeted table in the URL, identified by the mandatory **sys_id** field.

Input Fields

All fields from the targeted table, excluding system fields, which will be used for updating the existing record. The **sys_id** field is used to locate the existing record.

Output Fields

Returns the **sys_id** of the record that was updated.

Sample SOAP Messages

Sample SOAP request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
    <soapenv:Header/>
    <soapenv:Body>
        <inc:update>
            <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>

            <short_description>this is updated</short_description>

        </inc:update>
    </soapenv:Body>
</soapenv:Envelope>

```

Sample SOAP response

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <updateResponse>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </updateResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

For language-specific **update** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

deleteRecord

Delete a record from the targeted table by supplying its **sys_id**.

Input Fields

An element **<sys_id>** identifying the **sys_id** of the record to be retrieved.

Output Fields

A **<count>** element within the **deleteRecordResponse** parent element indicating the number of records deleted, this will always equal to "1" for **deleteRecord**.

Sample SOAP Messages

Sample SOAP request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:deleteRecord>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </inc:deleteRecord>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <deleteRecordResponse>
      <count>1</count>
    </deleteRecordResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

For language-specific **deleteRecord** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

deleteMultiple

Delete multiple records from the targeted table by example values.

Input Fields

All fields from the targeted table, including system fields, are used in query-by-example (QBE) fashion to locate records to be deleted. Query example fields can have special prefixes to constrain the search function.

Output Fields

A **<count>** element within the deleteRecordResponse parent element indicating the number of records deleted

Sample SOAP Messages

Sample SOAP request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:deleteMultiple>
      <category>hardware</category>
    </inc:deleteMultiple>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <deleteMultipleResponse>
      <count>6</count>
    </deleteMultipleResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

For language-specific **deleteMultiple** examples, see:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

References

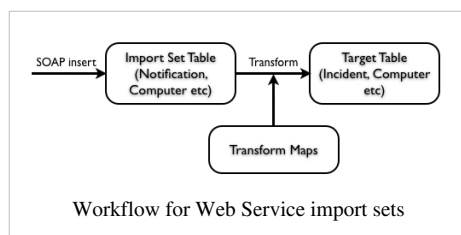
- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/web-services-apis/reference/r_DirectWebServiceAPIFunctions.html

Web Service Import Sets



Note: This article applies to Fuji and earlier releases. For more current information, see *Web Service Import Sets* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview



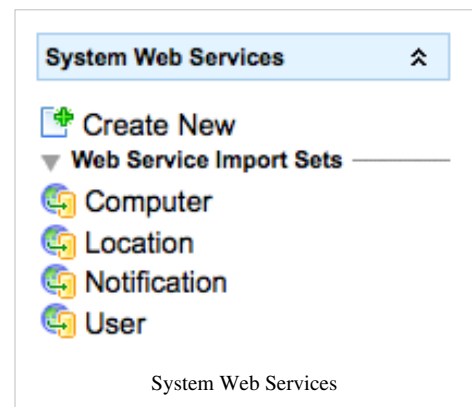
Web service import sets provides a web service interface for import set tables. By default, this type of web service will transform the incoming data synchronously based on the associated transform maps. If the associated import set mode is set to *Asynchronous*, the behavior is to save the data for transformation at a later time. Web service import sets tables publish all the default web service functions in the WSDL ^[4].

This plugin also provides these standard import set tables:

- Computer [imp_computer]
- Location [imp_location]
- Notification [imp_notification]
- User [imp_user]

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

http://<instance name>.service-now.com/imp_notification.do?WSDL



Security Requirements

Web service import sets use the same security mechanisms as SOAP web services or RESTful web services, depending on which protocol you use:

- Basic authentication requires a web service user provide a valid user name and password
- Contextual security requires a web service user meet the queried table's access control rules

Roles

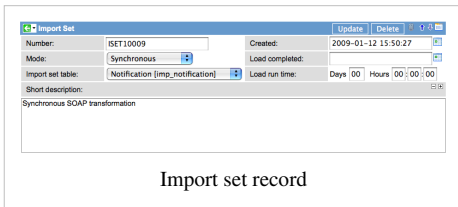
Certain roles are required for using web service import sets.

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.

If your instance uses high security settings, the web service user may also need the **soap** role.

Import Set Mode

When a web service message inserts a record into an import set table, and the table did not previously have an import set referenced, a new one will be created, and its **Mode** will be set as **Synchronous**. A synchronous import set transforms the data as soon as it is inserted, provided that the transform map already exists. This import set will also have a default **State** of **Loading**. By default, the state of all synchronous import sets is automatically set to **Processed** at midnight. As a result, when a new insert happens to the same table, a new synchronous import set will be created.



Import set record

Changing this import set to a **Mode** of **Asynchronous** and a **State** of **Loading** prevents the data from being transformed immediately. Using this configuration, the import set is loaded but not transformed. You can then perform the transformation manually or with a scheduled script job.

Mode	State	Function
Asynchronous	Loading	Data transformation does not occur automatically. Data added to import set row has a State of Pending . Data transformation can be scheduled or performed manually after the state changes to Loaded .
Asynchronous	Loaded	Marks the completion of data loading. Data transformation can now occur in a scheduled fashion or manually.
Synchronous	Loading	Data transformation occurs automatically and immediately whenever data is inserted into the associated import set row.
Synchronous	Loaded	When new data is inserted into this associated import set, a new import set with a Mode of Synchronous and State of Loading is created. Changing the state to Loaded indicates that a new synchronous import set should be created for the next import set row insert and transformed immediately.

Controlling Insert Behavior

In imports sets that specify one or more coalesce fields, records with a matching coalesce value are transformed from source to target table serially (one at a time) to prevent duplicates, starting with the Fuji release, Eureka Patch 7, and Dublin Patch 7.

In import sets that do not specify any coalesce field, records are transformed concurrently. You can control this behavior using the `glide.import_set_insert_serialized_when_no_coalesce` property. This property is **true** by default for instances that upgrade to one of these releases, **false** for new instances starting with one of these releases.

The system property `glide.soap.import_set_insert_serialized.<table name>`, controls how the instance inserts records from web service calls into a specific import set table. When true, this property prevents identical simultaneous inserts from creating duplicate records by serializing the database insert operations. If a target table does not have any coalesce fields defined in a transform map, set this property to **false** to improve web service

import set performance.

The property `glide.import_set_insert_serialized.<table name>` provides the same functionality, starting with the Fuji release, Eureka Patch 7, and Dublin Patch 7.



Note: *Disabling serialization can result in the creation of duplicate records when a coalesce field is defined for a target table transform map.*

See Available System Properties for detailed information about these properties.

Controlling Maximum Request Body Size

You can limit the maximum size allowed in a request body. By default, the maximum size allowed is 70MB. This maximum size applies to all SOAP requests, but usually only insert or insertMultiple operations reach this limit.

To change the maximum request body size, navigate to **System Web Services > Properties** and change the value of the **Maximum total size allowed in the SOAP request body, in MB** property. Specify a value of **-1** for no limit.



Note: *Setting the maximum higher than 70MB can cause the instance to run out of memory during very large imports.*

Controlling insertMultiple WSDL Format

When inserting records using SOAP web service import sets, the instance generates a WSDL that describes the inserted records. The property `glide.wsdl.consistent_insert_multiple` controls the format of the WSDL used by the insertMultiple operation, starting with the Fuji release. When enabled, this property causes the insertMultiple WSDL format to be consistent with the SOAP response and with other web service import set WSDLs. Prior to Fuji, the insertMultiple WSDL format is not consistent with the SOAP response. Enable this property when using insertMultiple if there is more than one transform map on the import set table.

If you activate the Insert Multiple Web Services plugin after upgrading to Fuji, this property is enabled by default. For instances using insertMultiple prior to Fuji, to enable the consistent WSDL format, navigate to **System Web Services > Properties** and select the **Make insertMultiple WSDLs consistent** property. See the web service import set WSDL examples for more information.

Standard Web Service Response

Inserting a record using a web service import set causes the instance to respond with information about the inserted record.

The response from a web service import set *insert* call returns the following response, depending on protocol.

SOAP Response

```

<SOAP-ENV:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <insertResponse>
      <sys_id>fa648f5f0a0a0b2b0048e7012448b8f1</sys_id>
      <table>incident</table>
      <display_name>number</display_name>
      <display_value>INC10014</display_value>
      <status>inserted</status>
    </insertResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Returned Values

The web service response returns the following field values, depending on the protocol used.

SOAP Field	Description
sys_id	The sys_id of the resulting record that was created or modified.
table	The table name of the table that was affected. In the case of an <i>Asynchronous</i> call, the table name would be the import set table such as <i>imp_notification</i> for the <i>Notifications</i> web service import set table.
display_name	The name of the field that is set as the display field for the record that was created or modified.
display_value	The value of the field designated as the display field. For example, the display field for the Incident table is the Number field and an example value is INC10001 .
status	A string value that indicates the action that occurred as a result of the web service invocation, relating to the record defined by the <i>sys_id</i> and <i>table</i> field values. <ul style="list-style-type: none"> inserted - the record was inserted updated - the record was updated ignored - the input was ignored, the record was not updated and no new record was created skipped - the input data was skipped (similar to ignored) due to missing coalesce values error - there was an error processing the input
status_message	This value translates to the value found in the <i>Comment</i> field of the import set row and usually contains information related to the <i>status</i> value eg. "No field values changed" when the status is "ignored". Setting this value to a customized string value will cause the SOAP response to contain an optional <i>status_message</i> field to be returned.
error_message	The message related to a status of <i>error</i> . When an error occurs, setting this value to a customized string value will cause the SOAP response to contain an optional <i>error_message</i> field to be returned.

Tailoring the SOAP Response

It is possible to include information other than the information specified in the WSDL by overwriting the contents of **status_message** using the transform script.

Editing a Web Service

Clicking on any of the web service import sets modules opens a form that allows you to edit the web service. You can also access this form using the **Edit Web Service** related link from any import set table list.

Editing Web Service

The **Name** of the web service is the table name of the import set table, which also appears in the **WSDL** field to display the URL to access the WSDL ^[4] definition for the web service.

Web Service Fields

The **Web Service Fields** related list displays the fields available for this web service. All fields by default are published as the XSD ^[2] type of *xsd:string*. The **Name** is the field that is exposed for the web service and therefore appears as the name of the field in the WSDL. The **Label** is the label of the field as it appears for the Import Sets.

You can add, delete or modify web service fields from this list.

Note: After modifying the list of web service fields, click **Update** to save the changes.



Web Service Transform Map

These are the transform maps associated with this web service. When the import set mode is *Synchronous*, the input data will be transformed immediately.

Creating a New Web Service

Navigate to **System Web Services > Create New**.

Creating a new Web Service

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

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

Web Service Fields

The fields available for this web service. All fields by default are published as the XSD ^[2] type of `xsd:string`. The *Name* is the field that is exposed for the web service and therefore appears as the name of the field in the WSDL. The *Label* is the label of the field as it appears for the import sets table.

You can *Add*, mark for *Delete* or modify (double click on the field) an existing web service field in this list.



Note: After adding web service fields, click **Create** to create the web service import set table.

To add other fields after the Web Service is created, find the target table, and add the fields to that table.

Mapping

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

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

Table Transform Map

Name: SOAP notification Created: 2008-12-17 14:39:29

Source table: notification [soap_notification] Target table: incident [incident]

Active: ☒ Run business rules: ☒ Enforce mandatory fields: (No) Order: 100

Copy empty fields: ☐ Run script: ☒

Script:

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

Update Delete

Related Links

Mapping Assist
Auto map matching fields

Field Maps

Source field	Target field	Coalesce
duration	calendar_duration	false
severity	severity	false
uuid	correlation_id	true
assignment_group	assignment_group	false

SOAP transform map

Adding Web Service Response Values

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

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

```
// create new elements called "transaction_id"
// and "hello" in the web service response
response.transaction_id = "abc123";
response.hello = "world";

status_message="message 1"; // this is the normal status_message
variable
```

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

SOAP

```

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

```

Inserting Multiple Records

You can insert multiple records in one SOAP request by using the insertMultiple operation. The insertMultiple operation is available for the Direct Web Service API and Web Service Import Sets. To enable insertMultiple, activate the Insert Multiple Web Service plugin.

Click the plus to expand instructions for activating a plugin.

If you have the admin role, use the following steps to activate the plugin.

1. Navigate to **System Definition > Plugins**.
2. Right-click the plugin name on the list and select **Activate/Upgrade**.

If the plugin depends on other plugins, these plugins are listed along with their activation status.

3. [Optional] If available, select the **Load demo data** check box.

Some plugins include demo data—sample records that are designed to illustrate plugin features for common use cases. Loading demo data is a good policy when you first activate the plugin on a development or test instance. You can load demo data after the plugin is activated by repeating this process and selecting the check box.

4. Click **Activate**.



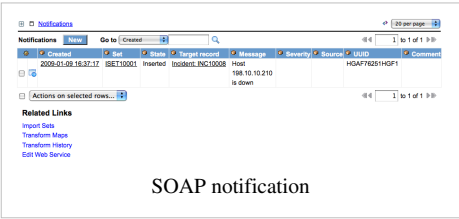
Note: Activating this plugin adds a new operation to the **WSDL**. After this plugin is activated, consume a new WSDL to update your web services client.

Administration

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

- **Import Sets:** the import sets related to this web service import set
- **Transform Maps:** a list of transform maps related to this web service
- **Transform History:** the transformation history
- **Edit Web Service:** edit the web service

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



Debugging

To debug incoming SOAP requests, you must add a system property.

Property	Description
glide.processor.debug.SOAPProcessor	Controls if the system logs all incoming SOAP requests. To prevent the system logs from growing unnecessarily, set this property to false when you finish debugging incoming web service requests. <ul style="list-style-type: none">• Type: true false• Default value: false• Location: Add to the System Property [sys_properties] table

Enhancements

Fuji

- You can control the WSDL format for the insertMultiple operation.
- You can specify the maximum body size for inbound SOAP requests.
- Insert operations to a given target table are performed concurrently for optimal performance when an import set has no coalesce field. The properties `glide.import_set_insert_serialized_when_no_coalesce` and `glide.import_set_insert_serialized.<table_name>` control how the instance inserts records from web service calls into an import set table.

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/import-sets/concept/c_WebServiceImportSets.html
[2] <http://www.w3.org/TR/xmlschema-2/>

Web Service Consumer



Note: This article applies to Fuji and earlier releases. For more current information, see *Outbound SOAP Web Service*^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The SOAP Message module can be used to develop, prototype, and save outbound SOAP messages that can be reused in business rules and scripts. Outbound SOAP web services comply with SOAP1.1 standards.

You can use outbound SOAP messages in scripts using the SOAPMessageV2 API and the SOAPResponseV2 API, starting with the Fuji release. Examples detailing how to script outbound SOAP are available. For versions prior to Fuji, refer to previous version documentation.

Video Tutorial

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

How to Consume Third-Party SOAP Web Services

SOAP Message

Information needed to send SOAP requests is stored in SOAP message records. Each record specifies an endpoint for the request, the required format of the request as a web services description language (WSDL) file, authentication information, and a list of functions that can run against the endpoint.

Creating a SOAP Message

1. Navigate to **System Web Services > SOAP Message**.
2. Click **New**.
3. Enter a **Name** to identify the SOAP message.
4. Specify a WSDL using one of these options:
 - To download and use an online WSDL source, select the **Download WSDL** check box and enter the URL for the WSDL in the **WSDL** field
 - To enter the WSDL directly, clear the **Download WSDL** check box, and then copy and paste the WSDL XML into the **WSDL XML** field.
5. If the endpoint is protected by basic authentication, select the **Use basic auth** check box and enter the credentials.
6. If the endpoint requires mutual authentication, select the **Enable mutual authentication** check box and select a **Protocol profile** to use for mutual authentication (starting with the Fuji release).
7. Click **Submit**.

This image shows an example of a SOAP message that connects to a demo instance of ServiceNow.

Related Links

[Generate sample SOAP messages](#)

SOAP Message Functions

SOAP Message Function

Update

Delete

Function:

insert

Basic auth user ID:

itil

Basic auth user password:

SOAP message:

demoi1 incident

Lock:

☒

Use basic auth:

☒

SOAP action:

http://www.service-now.com/incident/insert

SOAP endpoint:

<https://demoi1.service-now.com/incident.do?SOAP>

Envelope: XML

```

<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tns="http://www.service-now.com/incident" xmlns:m="http://www.service-now.com" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <insert xmlns="http://www.service-now.com">
      <short_description xsi:type="xsd:string">${short_description}</short_description>
    </insert>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Update

Delete

SOAP Message Parameters	
New SOAP Function = insert	
Name	Value
short_description	this is the short description

The **SOAP action**, **SOAP endpoint**, and **Envelope** fields should be populated automatically based on the WSDL definition. The **Envelope** defines the message to send to the endpoint. In this example, the **Envelope** values have this format:

```
...
<!-- optional -->
<short_description xsi:type="xsd:string">String</short_description>
...
```

To submit a specific value, enter the value directly in the appropriate XML tag. In this example, to set the **Short description** for a record, enter:

```
...
<short_description xsi:type="xsd:string">This is the short description</short_description>
...
```

Variable Substitution

To use variable substitution, use the format `${<variable_name>}` instead of defining a specific value.

```
...
<short_description xsi:type="xsd:string">${short_desc}</short_description>
...
```

To test variable substitution after you have modified the SOAP envelope with the variables, define values for the variables in the **SOAP Message Parameters** related list. For example, click **New** and enter the following information:

SOAP Message Parameters		Update	Delete			
Name:	short_desc	SOAP Function:	insert			
Value:	this is the short descript					
Update		Delete				

Testing

To test the SOAP message, click the **Test** related link. You are redirected to a test result form as shown below.

SOAP Message Test [Update] [Delete]

SOAP Function: Created:

Request: **XML**

```
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tns="http://www.service-
now.com/incident" xmlns:m="http://www.service-now.com" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-
```

HTTP Status:

Response: **XML**

```
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:m="http://www.service-now.com"
xmlns:tns="http://www.service-now.com/incident"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
```

[Update] [Delete]

Related Links

[Rerun test](#)

You can see the original SOAP request message, the resulting HTTP status code, and the SOAP response in this screen. You can also click the **Rerun test** related link to resubmit the SOAP request.



Note: A test SOAP message will time out after 60 seconds if a response is not received.

Using a SOAP Message in a Script

After you have developed and tested the SOAP message, click the **Preview script usage** related link in the SOAP Message Function form. The dialog box displays an example of how you can invoke the SOAP message from a script. See SOAPMessageV2 API for a list of available methods, or Scripting Outbound SOAP for detailed scripting examples.

Preview SOAP message script usage [X]

```
try {
  var s = new SNC.SOAPMessageV2('StockQuote', 'StockQuoteSoap.GetQuote');
  s.setStringParameterNoEscape('symbol', 'NOW');
  var response = s.execute();
  var responseBody = response.getBody();
  var status = response.getStatusCode();
}
catch(ex) {
  var message = ex.getMessage();
}
```

You can manipulate the resulting XML response body with `XMLDocument` or with `gs.getXMLText` and `gs.getXMLNodeList`.

Demonstration Video

The following video shows examples of creating and sending SOAP messages. The script example in this video uses the version one SOAPMessage API.

Sending a SOAP Message Through a MID Server

When creating SOAP message functions, you can configure the function to be sent through a MID Server selected in the **Use MID Server** field. There must be a running MID Server associated with your ServiceNow instance to use this functionality. All SOAP messages sent through a MID Server are performed asynchronously.

SOAP Message Function | = Required field [Update] [Delete]

Function: TempConvertSoap.FahrenheitToC

Use MID server: win2008server

SOAP message: TemperatureConvert

Lock: ☒

Use basic auth: ☐

Use WS-Security: ☐

Strip whitespace: ☐

SOAP action: http://tempuri.org/FahrenheitToCelsius

SOAP endpoint: http://www.w3schools.com/webservices/tempconvert.asmx

Envelope: XML

```
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:m="http://tempuri.org/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <FahrenheitToCelsius xmlns="http://tempuri.org/">
      <!-- optional -->
      <Fahrenheit xsi:type="xsd:string">${temperatureF}</Fahrenheit>
    </FahrenheitToCelsius>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

By specifying a MID Server, all SOAP requests that use this SOAP message are sent through that MID Server. You can override the selected MID Server by using the `setMIDServer(mid server)` API call in a script.

Connectivity Details

For the ServiceNow-initiated SOAP requests to successfully communicate with the web service provider inside a remote network, the ServiceNow instance must have HTTP or HTTPS access to the SOAP endpoint at the provider. Like any integration, such as LDAP, web services, or JDBC, the SOAP endpoint may reside behind a firewall that is blocking inbound communication from the ServiceNow instance. If this is the case, you need to make network changes to allow this connectivity into your network. You can either modify the firewall and ACL rules to allow the ServiceNow IP address, configure the SOAP message to use a MID Server, or implement a VPN tunnel to allow the ServiceNow communication into your network.



Note: A common misconception is that because asynchronous SOAP requests are routed through the ECC queue, they are always sent through a MID Server. This is not the case. Asynchronous SOAP requests only use a MID Server when configured to do so.

Outbound SOAP Security

You can authenticate outbound SOAP messages using several different security protocols. The security protocol you should use depends on the requirements of the web service provider.

Basic Authentication

If the endpoint requires a user name and password, you can provide these credentials using basic authentication.

1. Navigate to **System Web Services > Outbound > SOAP Message**.
2. Select a SOAP message record.
3. In the **SOAP Message Functions** related list, select a function.
4. Select **Use basic auth**.
5. Enter a user name in the **Basic auth user ID** field.
6. Enter the password for that user in **Basic auth user password**.
7. Click **Update**.

Web Service Security

You can sign outbound SOAP messages using a key store and trusted server certificate saved on the instance.

1. Upload a key store certificate with a **Type** of **Java Key Store** or **PKCS12 Key Store**.
2. Upload the trusted server certificate for the key store certificate.
3. Navigate to **System Web Services > Outbound > SOAP Message**.
4. Select a SOAP message record.
5. In the **SOAP Message Functions** related list, select a function.
6. Select **Use WS-Security**.
7. In the **Key store** field, select the Java or PKCS12 key store you uploaded.
8. In the **Key store alias** field, enter the alias that identifies the public and private key pair.
9. In the **Key store password** field, enter the password you assigned the key store record.
10. In the **Certificate** field, select the trusted certificate for the selected key store.
11. Click **Update**.

Mutual Authentication

ServiceNow supports mutual authentication for outbound web services. Mutual authentication is not available for inbound web services.

Configuring SOAP with a Proxy

The following properties provide support for SOAP requests to use a web proxy server.

Property	Description	Examples
glide.http.proxy_host	The proxy server hostname or IP address	proxy.company.com, 192.168.34.54
glide.http.proxy_port	The port number for the proxy server	8080, 9100
glide.http.proxy_username	If the proxy server is authenticating using user name and password, enter a value for this property	proxyuser
glide.http.proxy_password	If the proxy server is authenticating using user name and password, enter a value for this property	password

Enhancements

Fuji

- The glide.outbound.sslv3.disabled system property can disable the SSLv3 protocol for outbound connections.
- Mutual authentication is available for outbound web services.
- The SOAPMessageV2 API and SOAPResponseV2 API provide enhanced interfaces for sending outbound SOAP messages using scripts.

References

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

Scripted Web Services



Note: This article applies to Fuji and earlier releases. For more current information, see *Scripted SOAP Web Services* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

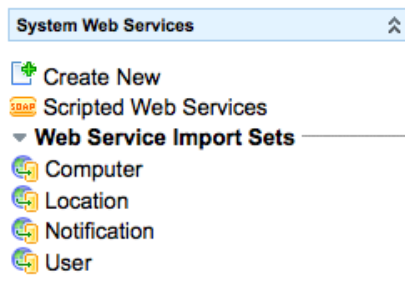
Scripted Web Services allow a ServiceNow administrator to create new web services that are not addressed by the system. You can define input and output parameters for the web service and use JavaScript to perform operations. Though this feature is very powerful, use Direct Web Services or Web Service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

Security

When strict security is enforced on a system, the HTTP authenticated user must have the **soap_script** role to execute the scripted web service.

Creating a new Web Service

When the Web Services Provider - Scripted plugin is activated, a new module **Scripted Web Services** is available under the **System Web Services** application.



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

sys_web_service New Go to Name 🔍			
	Name	Active	Short description
<input type="checkbox"/>	GetProperty	true	Get a property value
<input type="checkbox"/>	GetTransactionCount	true	Get the number of transactions
<input type="checkbox"/>	OrderBlackBerry	true	Order a BlackBerry
<input type="checkbox"/> Actions on selected rows... ⬆️⬆️⬆️			

Example 1: Retrieving a System Property

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

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

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

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

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

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

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

The output parameters are set by using the following syntax:

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

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

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

Example 2: Ordering a Blackberry

Direct Web Services in ServiceNow operate on tables and their data, while the following example shows how to initiate a business solution, such as ordering a Blackberry, by invoking a scripted Web Service. The following input and output parameters will support the Blackberry example:

Input Parameters		New
		Name
<input type="checkbox"/>		phone_number
<input type="checkbox"/>		requested_for
<input type="checkbox"/>	Actions on selected rows...	

Output Parameters		New
		Name
<input type="checkbox"/>		request_number
<input type="checkbox"/>	Actions on selected rows...	

This script shows how to use the above parameters to add a Blackberry to the service catalog shopping cart and order it. The request number is returned in the **request_number** field of the SOAP response.

```

var cart = new Cart();
var item = cart.addItem('e2132865c0a8016500108d9cee411699');
cart.setVariable(item, 'original', request.phone_number);

// set the requested for
var gr = new GlideRecord("sys_user");
gr.addQuery("user_name", request.requested_for);
gr.query();
if (gr.next()) {
    var cartGR = cart.getCart();
    cartGR.requested_for = gr.sys_id;
    cartGR.update();
}

var rc = cart.placeOrder();
response.request_number = rc.number;

```

Global Variables

To facilitate custom processing of incoming SOAP requests, the following global variables are available in the script context:

1. **soapRequestDocument**: a Java *org.w3c.dom.Document* object representing the incoming SOAP envelope.
2. **soapRequestXML**: a string object representing the incoming SOAP envelope XML.
3. **request**: a Javascript object containing mapped values (mapped to input parameter names) of the incoming SOAP envelope
4. **response**: a Javascript object which allows the script programmer to customize the response values. See Customized Response

Customized Response

To customize and have control over the XML payload of the SOAP response, follow this example:

1. Create a customized XML document using the XMLDocument script include object
2. set its document element to the variable **response.soapResponseElement** in a scripted web service

For example, the following scripted web service script:

```

var xmldoc = new XMLDocument("<myResponse></myResponse>");

xmldoc.createElement("element_one", "test"); // creates the new element at the document element level if setCurrent is never called
xmldoc.createElement("element_two", "new2 value"); // calling without a value will create a new element by itself

var el = xmldoc.createElement("element_three");

xmldoc.setCurrent(el); // this is now the parent of any new elements created subsequently using createElement()

xmldoc.createElement("newChild", "test child element");

response.soapResponseElement = xmldoc.getDocumentElement();

```

Is used to accept the following request:

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:tes="http://www.service-now.com/TestCustomResponse">
  <soapenv:Header/>
  <soapenv:Body>
    <tes:execute/>
  </soapenv:Body>
</soapenv:Envelope>
```

Which will respond with the following SOAP response:

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:tes="http://www.service-now.com/TestCustomResponse">
  <soapenv:Header/>
  <soapenv:Body>
    <myResponse>
      <element_one>test</element_one>
      <element_two>new2 value</element_two>
      <element_three>
        <newChild>test child element</newChild>
      </element_three>
    </myResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

WSDL support will need to be created externally. The SOAP endpoint will need to be referred back to the scripted web service in question.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-soap/concept/c_ScriptedWebServices.html

Hierarchical Web Service

Overview

Support is available for inserting hierarchical data into tables or web service import set tables. The hierarchical data in the **Insert** API is automatically mapped to related records of the targeted table.

To enable hierarchical SOAP insert globally, by creating and setting the property `glide.web_service.hierarchical` to "true". The client of the API can also override this value by invoking the SOAP web service with the URL parameter **hierarchical=true**.

By default, hierarchical web services limit the hierarchy to a maximum of 3 levels. You can control this limit using the property `glide.wsdll.maximum_hierarchy`. For optimal performance, do not set this property to a value higher than 3.

Example

For example, when a related list is created for the incident table called "u_custom_comments"

Incident [Self Service view] Update Close Incident Delete

Number: INC0010001 Opened: 2010-03-21 21:30:33

Caller: Closed:

Short description: test hierarchical

Additional comments: ABC

Update Close Incident Delete

Custom Comments New + - Incident = INC0010001 1 to 1 of 1	
Comment	Comment Type
comment 1	travel

And "u_comment_items" is created as a related list for "u_custom_comments"

Custom Comments [Self Service view] Update Delete

Comment: comment 1

Comment Type: travel

Incident: INC0010001

Update Delete

Comment Items New + - Comment = 8422ebefc0a8006e2e1c9760289f9dbb 1 to 1 of 1	
Value	Name
value 1	name 1

SOAP

When the SOAP message is constructed from the hierarchical web service described by the WSDL above and invoked, it will create the "incident", "u_custom_comments", and "u_comment_items" records.

```
https://instance.service-now.com/incident.do?SOAP&hierarchical=true
```

Request

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:insert>
      <short_description>test hierarchical</short_description>
      <u_custom_comments>
        <u_comment>comment 1</u_comment>
        <u_comment_type>travel</u_comment_type>
        <u_comment_items>
          <u_name>name 1</u_name>
          <u_value>value 1</u_value>
        </u_comment_items>
      </u_custom_comments>
    </inc:insert>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <insertResponse>
      <sys_id>8422ebe7c0a8006e7d23848c2dc8ba47</sys_id>
      <number>INC0010001</number>
    </insertResponse>
  </soapenv:Body>
</soapenv:Envelope>
```


Attachment Creator Web Service



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

Overview

Attaching documents to records in ServiceNow is achieved by sending a SOAP message targeting the **ecc_queue** table. For example, using the following URL or target end point:

https://instance_name.service-now.com/ecc_queue.do?SOAP

The following fields in **ecc_queue** must be set to trigger the creation of the attachment.

Field Name	Description	Value
agent	The name of the agent sending in the request, this can be any value since it is not used for processing.	AttachmentCreator
topic	The topic of the queue record, this value must be set to "AttachmentCreator" to trigger the attachment creation	AttachmentCreator
name	This field must contain a ":" delimited value of the file name, and its content-type	file_name.xls:application/vnd.ms-excel
source	This field must contain a ":" delimited value of the target table and its sys_id	incident:dd90c5d70a0a0b39000aac5aee704ae8
payload	This field must contain the Base 64 ^[2] encoded string representing the object to be attached	the base 64 encoded string

Sending in the values described in the table above will attach an Excel file to the **incident** table for the record located by the **sys_id** "dd90c5d70a0a0b39000aac5aee704ae8"

Security

Like all other HTTP based web services available on the platform, the AttachmentCreator SOAP web service is required to authenticate using basic authentication ^[3] by default. The user ID that is used for authentication will be subjected to access control in the same way as an interactive user.

To create attachments, the SOAP user must have any roles required to create Attachment [sys_attachment] records, as well as the soap_create role, and any roles required to read and write records on the target table, such as the itil role to add attachments to incident records. By default there is no single role allowing you to add attachments. You can create a role to explicitly allow adding attachments, then assign this role to the SOAP user.

File Type Security

You can control what file types users can attach by setting the `glide.attachment.extensions` and `glide.security.file.mime_type.validation` properties. These properties apply to the AttachmentCreator web service starting with Fuji Patch 10.

For these properties to apply to the AttachmentCreator web service, the property `glide.attachment.enforce_security_validation` must be set to `true`. This property is `true` by default.

Example SOAP Message

The following is an example of a SOAP message that would take a text file "john1.txt" of mime-type: text/plain and attach it to an Incident with a GUID of: e6eed6950a0a3c59006f32c8e3ff3cf9. Note the payload is the base64 encoding of the file itself.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ecc="http://www.service-now.com/ecc_queue">
  <soapenv:Header />
  <soapenv:Body>
    <ecc:insert>
      <agent>AttachmentCreator</agent>
      <topic>AttachmentCreator</topic>
      <name>john1.txt:text/plain</name>
      <source>incident:e6eed6950a0a3c59006f32c8e3ff3cf9</source>
      <payload>SSB3b25kZXIgaWYgc2hlIGtub3ducyB3aGF0IHNoZSdzIGRvaW5nIG5vdy4K</payload>
    </ecc:insert>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Node.js Script

The following example Node.js ^[4] script adds an attachment to an incident record. Run this script from a client computer, not a ServiceNow instance.

```
/**
 *
 * Node.js to ServiceNow attachment upload via SOAP
 *
 * Andrew Venables andrew.venables@servicenow.com
 * July 2014
 *
 * Version 1.0
 *
 */

var soap = require('soap'), // https://github.com/vpulim/node-soap
    mime = require('mime'), // https://github.com/broofa/node-mime
    fs = require("fs");
```

```
var WSDL_FILENAME = 'ecc_queue.xml'; // Goto
https://instancename.service-now.com/ecc_queue.do?WSDL and save a copy
of the WSDL locally for simplicity
var DIRECTORY_CONTAINING_FILES =
'/Users/andrew.venables/Documents/Uploads'; // Local path to the
directory containing all the files we want to upload
var USERNAME = 'andy.venables'; // An admin user account on the
instance
var PASSWORD = 'MY_PASSWORD'; // Password for above account
var TARGET_TABLE = 'incident'; // Target table to attach the files to
var TARGET_SYS_ID = '9d385017c611228701d22104cc95c371'; // Target
record / sys_id to attach the files to. OOTB INC0000002

var files_to_upload; // Global that will contain our list of files to
be uploaded
var pos = 0; // Global pointer for our position in the files_to_upload
list

// Create a SOAP client to use to post to the instance
soap.createClient(WSDL_FILENAME, function(err, client) { // Node uses
callbacks
    if (err) console.error(err);

    // Set the username and password
    client.setSecurity(new soap.BasicAuthSecurity(USERNAME,
PASSWORD));

    // Read all the files in our source directory, will include . and
..
    files_to_upload = fs.readdirSync(DIRECTORY_CONTAINING_FILES);

    console.log('Files to upload: ' + files_to_upload.length + '\n');

    // Start iterating through the list of files to upload
    next(client);
});

// Process the next file in the files_to_upload array
// This is a neat way of dealing with Node and its expectation of
callbacks
function next(client) {

    // Check we haven't reached the end
    if (pos >= files_to_upload.length) return;

    // Get the next file to upload
```

```
var file_name = files_to_upload[pos];

// Increment the pointer to the next file
pos++;

console.log(pos + '/' + files_to_upload.length+ ' - Uploading
file: ' + file_name);

// A blank file is the end of the list
if (file_name == '') return;

// Skip to the next file as this one is invalid
if (file_name == '.' || file_name == '..' ||
file_name.indexOf('.') == 0)
    next(client);

// Get the file type using an module called mime
var file_type = mime.lookup(file_name);
console.log('    of type: ' + file_type);

var file_payload;
// Load the file into a buffer
fs.readFile(DIRECTORY_CONTAINING_FILES + '/' + file_name,
function(err, the_data) {
    if (err) console.error(err);

    // Encode the buffer to base64
    file_payload = new Buffer(the_data,
'binary').toString('base64');

    // Set the parameters before we call the Web Service
    var parameters = {
        'agent':      'AttachmentCreator',
        'topic':      'AttachmentCreator',
        'name':       file_name+':'+file_type,
        'source':     TARGET_TABLE+':'+TARGET_SYS_ID,
        'payload':    file_payload
    };

    console.log('    sending...')
    // Make the Web Service call, remember node likes callbacks
    client.insert(parameters, function(err, result) {
        if (err) console.error(err);

        console.log(result);

        // This file is done, next!
```

```

        next(client);
    });
});
}

```

Example Perl Script

The following example inserts multiple attachments into the ecc_queue table. You can specify the directory to upload using the file_path argument. All files in that directory and all subdirectories are uploaded.

This script requires a before business rule on inserts to the ecc_queue table to avoid having to make multiple calls for each attachment import.

```

use warnings;
use File::Basename;
use File::Find qw(finddepth);
use ServiceNow::SOAP;
use MIME::Base64;
use MIME::Types;

# Handle command line arguments to get the destination folder
$numArgs = $#ARGV + 1;
if( $numArgs != 6)
{
    print "ERROR: Insufficient Command Line Arguments.\n\n";
    print_usage();
    exit -1;
}

$SNC_HOST = $ARGV[0];
$username = $ARGV[1];
$password = $ARGV[2];
$file_path = $ARGV[3];
$target_table = $ARGV[4];
$correlation_field = $ARGV[5];

my $base64;
my $buf;
my $mimetype;

#SOAP CONFIG stuff
my $CONFIG = ServiceNow($SNC_HOST, $username, $password);
my $sn_table = $CONFIG->table('ecc_queue');

# traverse the specified file directory and stores files in the @files
array
my @files;
finddepth(sub {
    return if($_ eq '.' || $_ eq '..');

```

```
    push @files, $File::Find::name;
}, $file_path);

# process all the files in our array
foreach $file (@files) {
    $full_path = "$file";
    print "\nFull Path: ". $file;

    #Test the file type
    if( -d $file)
    {
        print "\nWe don't send directories only files: ".
$full_path;
        next;
    }

    open( FILE, $file) or die "$!";
    binmode FILE; #preserves file formatting on Windows
    my $file_contents = "";
    while ( read( FILE, $buf, 60 * 57 ) ) {
        $file_contents .= encode_base64($buf);
    }

    my $filename = basename($full_path);
    my $mimetype = MIME::Types -> new;

    #Get Target Record by Correlation Field - $number should retrieve the
    matching correlation value from the filename
    # In this case, the correlation value is the first 5 characters of
    the filename
    my $number = substr $filename, 0, 5;

    my $sysid = $sn_table->insert(
        agent => 'AttachmentCreator',
        topic => 'AttachmentCreator',
        name => $filename.'.'.$mimetype->mimeTypeOf($filename),
        source => $target_table.':ka'.$number.':'.$correlation_field,
        payload => $file_contents
    );

    # print results or faults
    print " Created Attachment record: " . $sysid . "\n";
}
```

```
# subroutine to print out the USAGE of this script
sub print_usage {
    print "=====\n";
    print "sn_attachment_import.pl\n";
    print "=====\n";
    print "\nUSAGE:\n\n";
    print "sn_attachment_import.pl <SN instanceUrl> <SN username> <SN password> <root_data_path>\n";
    print "\n...Where:\n";
    print "\t<SN instance name>: The name of the target ServiceNow instance, i.e.
instance (for instance.service-
now.com)\n";
    print "\t<username>: ServiceNow user name with rights to add records to the
target table\n";
    print "\t<password>: ServiceNow user password\n";
    print "\t<root_data_path>: The full path and name of the directory that contains
the attachment data.\n";
    print "\t<target_table>: The ServiceNow table the attachment should be
associated with.\n";
    print "\t<correlation_field>: The ServiceNow field on the target table that matches
the correlation value of the attachments\n";
    print "\nEXAMPLE:\n\n";
    print "sn_attachment_import.pl myinstancename myusername
mypassword ./attachment_root_folder sn_table_name correlation_field\n";
    print "\n-----\n\n";
}
```

Before Business Rule

Create the following before business rule to manage inserting multiple attachments.

Field	Value
Table	ecc_queue
When	Before
Insert	true
Condition	current.source.indexOf(':ka') > -1

Script

```
onBefore(current, previous) {  
    //This function will be automatically called when this rule is  
    processed.  
    var log = [];  
    log.push('Starting Attachment Correlation\nAgent: '+  
current.agent);  
    var correlation_field =  
current.source.split(':ka')[1].split(':')[1]+'';  
    var correlation_id =  
current.source.split(':ka')[1].split(':')[0]+'';  
    var target_table = current.source.split(':ka')[0]+'';  
    log.push('Field: '+ correlation_field +'\nID: '+ correlation_id  
+ '\nTable: '+ target_table);  
    var gr = new GlideRecord(target_table);  
    if(correlation_field != '' && correlation_id != ''){  
        gr.addQuery(correlation_field, correlation_id);  
        gr.query();  
        gr.next();  
        current.source = target_table + ':' + gr.sys_id;  
        log.push('Source: '+ current.source);  
    }  
    gs.log(log.join('\n'));  
}
```

FAQ

- How many attachments can you send in one message
 - Only one attachment per message
- Is there size limit on the attachment in the message
 - Attachments are limited to a maximum of 5MB.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-soap/reference/r_AttachmentCreatorSOAPWebService.html
- [2] <http://en.wikipedia.org/wiki/Base64>
- [3] <http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html#BasicAA>
- [4] <http://nodejs.org/>

JSON Web Service

JSON



Note: This article applies to Fuji and earlier releases. For more current information, see *JSONv2 Web Service* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The ability to describe sets of data in JSON ^[2] format is a natural extension to the JavaScript language. ServiceNow supports a web service interface that operates on the JSON object as the data input and output format.



Note: The JSON Web Service feature is replaced by JSONv2 starting with the Dublin release.

The JSON web service is provided by a platform-level processor similar to the services for SOAP, WSDL, CSV, Excel, and XML. Like those services, the JSON service is triggered by the standalone JSON URL parameter. For example:

```
https://<instance name>.service-now.com/mytable.do?JSON
```

Having the JSON object available as a data format for web services means that you can create (insert), update, and query any data in the ServiceNow platform using the JSON object format, and get results in the JSON object format.

Security

Like all other HTTP-based web services available on the platform, the JSON web service is required to authenticate using basic authentication ^[3] by default. The user ID that is used for authentication is subjected to access control in the same way as an interactive user.

Activating the Plugin

An administrator must activate the JSON Web Service plugin to configure a JSON web service.

Click the plus to expand instructions for activating a plugin.

If you have the admin role, use the following steps to activate the plugin.

1. Navigate to **System Definition > Plugins**.
2. Right-click the plugin name on the list and select **Activate/Upgrade**.

If the plugin depends on other plugins, these plugins are listed along with their activation status.

3. [Optional] If available, select the **Load demo data** check box.

Some plugins include demo data—sample records that are designed to illustrate plugin features for common use cases. Loading demo data is a good policy when you first activate the plugin on a development or test instance. You can load demo data after the plugin is activated by repeating this process and selecting the check box.

4. Click **Activate**.

JSON Object Format

The JSON object is built in two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

In its simplest form, a JSON object is just a comma delimited set of name/value pairs. For example:

```
{"name one": "value one", "name two": "value two"}
```

The following is a sample of a single record array of incidents in JSON:

```
{ "records":  
  [{ "closed_by": "",  
    "__status": "success",  
    "category": "inquiry",  
    "escalation": "0",  
    "state": "1",  
    "location": "",  
    "reassignment_count": "0",  
    "time_worked": "",  
    "order": "0",  
    "due_date": "",  
    "number": "INC0010180",  
    "upon_approval": "proceed",  
    "sla_due": "2010-03-04 22:51:49",  
    "follow_up": "",  
    "notify": "1",  
    "business_stc": "0",  
    "caused_by": "",  
    "rejection_goto": "",  
    "assignment_group": "d625dccec0a8016700a222a0f7900d06",  
    "incident_state": "1",  
    "opened_at": "2010-02-23 22:51:49",  
    "wf_activity": "",  
    "calendar_duration": "",  
    "group_list": "",  
    "caller_id": "",  
    "comments": "",  
    "priority": "3",  
    "sys_id": "fd0774860a0a0b380061bab9094733ad",  
    "sys_updated_by": "itil",  
    "variables": "",  
    "delivery_task": "",  
    "sys_updated_on": "2010-02-23 22:51:49",  
    "parent": "",  
    "active": "true",  
    "opened_by": "681b365ec0a80164000fb0b05854a0cd",  
    "expected_start": ""
```

```

    "sys_meta": "System meta data",
    "watch_list": "",
    "company": "",
    "upon_reject": "cancel",
    "work_notes": "",
    "sys_created_by": "itil",
    "cmdb_ci": "",
    "approval_set": "",
    "user_input": "",
    "sys_created_on": "2010-02-23 22:51:49",
    "contact_type": "phone",
    "rfc": "",
    "approval_history": "",
    "activity_due": "",
    "severity": "3",
    "subcategory": "",
    "work_end": "",
    "closed_at": "",
    "close_notes": "",
    "variable_pool": "",
    "business_duration": "",
    "knowledge": "false",
    "approval": "not requested",
    "sys_mod_count": "0",
    "problem_id": "",
    "calendar_stc": "0",
    "work_start": "",
    "sys_domain": "global",
    "sys_response_variables": "",
    "correlation_id": "",
    "sys_class_name": "incident",
    "short_description": "this was inserted with python",
    "impact": "1",
    "description": "",
    "correlation_display": "",
    "urgency": "3",
    "assigned_to": "",
    "made_sla": "true",
    "delivery_plan": ""
  }
}

```

The following is a record array of incident responses with an error.

```

{
  "records": [
    {
      "__error": {

```

```
    "message": "Invalid Insert into: incident",
    "reason": "Data Policy Exception:  Short description is mandatory "
  },
  "__status": "failure",
  "active": "true",
  "activity_due": "",
  "approval": "not requested",
  "approval_history": "",
  "approval_set": "",
  "assigned_to": "",
  "assignment_group": "d625dccec0a8016700a222a0f7900d06",
  "business_duration": "",
  "business_stc": "",
  "calendar_duration": "",
  "calendar_stc": "",
  "caller_id": "",
  "category": "inquiry",
  "caused_by": "",
  "child_incidents": "0",
  "close_code": "",
  "close_notes": "",
  "closed_at": "",
  "closed_by": "",
  "cmdb_ci": "",
  "comments": "",
  "comments_and_work_notes": "",
  "company": "",
  "contact_type": "phone",
  "correlation_display": "",
  "correlation_id": "",
  "delivery_plan": "",
  "delivery_task": "",
  "description": "",
  "due_date": "",
  "escalation": "0",
  "expected_start": "",
  "follow_up": "",
  "group_list": "",
  "impact": "3",
  "incident_state": "1",
  "knowledge": "false",
  "location": "",
  "made_sla": "true",
  "notify": "1",
  "number": "INC0010001",
  "opened_at": "2013-07-23 18:01:17",
  "opened_by": "6816f79cc0a8016401c5a33be04be441",
```

```
    "order": "",
    "parent": "",
    "parent_incident": "",
    "priority": "5",
    "problem_id": "",
    "reassignment_count": "0",
    "reopen_count": "0",
    "resolved_at": "",
    "resolved_by": "",
    "rfc": "",
    "severity": "3",
    "short_description": "",
    "skills": "",
    "sla_due": "",
    "state": "1",
    "subcategory": "",
    "sys_class_name": "incident",
    "sys_created_by": "admin",
    "sys_created_on": "2013-07-23 18:01:17",
    "sys_domain": "global",
    "sys_id": "a96479343cb60100a92ec9a477ba9e45",
    "sys_mod_count": "0",
    "sys_updated_by": "admin",
    "sys_updated_on": "2013-07-23 18:01:17",
    "time_worked": "",
    "upon_approval": "proceed",
    "upon_reject": "cancel",
    "urgency": "3",
    "user_input": "",
    "watch_list": "",
    "work_end": "",
    "work_notes": "",
    "work_notes_list": "",
    "work_start": ""
  }
]
}
```

Response Status

In both of the code samples, the JSON object returned elements that describe the status of the GlideRecord response and whether the response included an error.

status

The `status` element uses this syntax:

```
"__status": "<value>",
```

where `<value>` is **success** or **failure**.

When the `status` element returns **failure**, the `error` element is added to identify the error and reason.

error

The `error` element uses this syntax:

```
"__error": {
  : "message": "<error value>",
  : "reason": "<reason value> "
  : }
```

where `<error value>` is the error message text and `<reason value>` is the reason the error was triggered.

Setting the Number of Rows Returned

The following system property controls how many rows JSON returns with each query.

Property	Description
<code>glide.processor.json.row_limit</code>	Specify the maximum number of rows a JSON query returns. <ul style="list-style-type: none"> • Type: integer • Default value: 250 • Location: Add to the System Properties [sys_properties] table

Action Parameters

Action parameters are separate and different from data parameters because they specify the action to take when the JSON object parameter is POSTed or when it is part of an HTTP GET. The parameters can also be specified as a field in the supplied JSON object. They have the effect of triggering an action in the case of `sysparm_action`, or filtering the results of an update or query in the case of `sysparm_query`.

sysparm_action

Following are the valid values for `sysparm_action` and the corresponding actions triggered by the API.

Data Retrieval

Method Summary	Description
<code>getKeys</code>	Query the targeted table using an encoded query string and return a comma delimited list of <code>sys_id</code> values.
<code>getRecords</code>	Query the targeted table using an encoded query string and return all matching records and their fields.
<code>get</code>	Query a single record from the targeted table by specifying the <code>sys_id</code> in the <code>sysparm_sys_id</code> URL parameter, and return the record and its fields.

Data Modification

Method Summary	Description
<code>insert</code>	Create one or more new records for the table targeted in the URL.
<code>insertMultiple</code>	Create multiple new records for the table targeted in the URL.
<code>update</code>	Update existing records for the table targeted in the URL, filtered by an encoded query string.
<code>deleteRecord</code>	Delete one record from the table targeted in the URL by specifying the <code>sys_id</code> in the <code>sysparm_sys_id</code> URL parameter.
<code>deleteMultiple</code>	Delete multiple records from the table targeted in the URL, filtered by an encoded query string.

sysparm_query

Specify an encoded query string to be used in `get`, `getRecords`, `update` or `deleteMultiple` `sysparm_action` values.

sysparm_view

Specify a form view to customize the return values for `get` and `getRecords` function calls. When a view is specified, the query returns only the fields defined in this view, including referenced values. If there is no view name, or if the view name is not valid, then the query returns all field names that are marked active in the dictionary.

sysparm_sys_id

Specify a target `sys_id` during a `get` or `delete` function call (`sysparm_action` value).

displayvariables

Set this boolean value to **true** during a `get` or `getRecords` function call to retrieve all variables attached to this record.

Data Retrieval API

You can query for data by issuing an HTTPS GET request to the platform. By default, a GET request is interpreted as a `get` function if a `sysparm_sys_id` parameter is present. Otherwise, it is interpreted as a `getRecords` function. You can also specify a URL parameter `sysparm_action=get`. Query responses are always encapsulated by a `records` hash of records, where each individual record's values are themselves hashed by field name.

Return Display Value for Reference Variables

When you are getting a record from a `get` or `getRecords` function, all the fields associated with that record are returned. The fields are often reference fields that contain a `sys_id` for another table. The base system behavior is to return the `sys_id` value for those fields. To have the display value for the field returned, use one of these options:

- Add the property `glide.json.return_displayValue` to the system properties, and every JSON request will return a display value for a reference field.
- Add the parameter `displayvalue=true` to the JSON request URL and JSON requests with that parameter will return a display value instead of the `sys_id` for a reference field. The JSON request URL would look like this:

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=getRecords&sysparm_query=active=true^category=hardware&displayvalue=true
```

- Add the parameter `displayvalue=all` to the JSON request URL, and JSON requests with that parameter will return a display value and the `sys_id` for a reference field. The response element name for the display value field will be prefixed with `dv_`, for example, `dv_caller_id`.

Return Display Variables

Use the `displayvariables` parameter to return an array of variables associated with a service catalog item record when using a `get` or `getRecords` function. The variables are expressed hierarchically. A variable container has the container's variables in its **children** field.

To get display variables, add the parameter `displayvariables=true` to the JSON request URL. The JSON request URL would look like this:

```
https://<instance name>.service-now.com/sc_req_item.do?JSON&sysparm_action=getRecords&sysparm_query=active=true^short_description=Laptop%20preconfigured%20for%20developers&displayvariables=true
```

White space has been added to the following example for clarity:

```
"variables": [{"children": null, "name": "asset_tag", "order": 1300, "question_text": "Asset Tag", "type": 6, "value": ""},
{"children": [{"children": null, "name": "ste1", "order": -1, "question_text": "Test", "type": 6, "value": "test"}], "name": "Test one", "order": -1, "question_text": "Testing", "type": 0, "value": null},
{"children": null, "name": "", "order": 200, "question_text": "Please specify an operating system", "type": 3, "value": "Windows XP"},
{"children": [{"children": null, "name": "create_item", "order": 1000, "question_text": "Create an Inventory Item?", "type": 7, "value": ""}], "name": null, "order": -1, "question_text": "Options", "type": 0, "value": null},
{"children": null, "name": "serial_number", "order": 1100, "question_text": "Serial Number", "type": 6, "value": ""},
{"children": null, "name": "hard_drive", "order": 100, "question_text": "What size hard drive do you want?", "type": 3, "value": "60 GB"},
{"children": null, "name": "name", "order": 1200, "question_text": "Name of new inventory item", "type": 6, "value": ""},
{"children": null, "name": "company", "order": 1400, "question_text": "Manufacturer", "type": 8, "value": ""}]
```

Controlling the Order of Records

You can control the order that records appear in the JSON response. To set an order, use the **ORDERBY** or **ORDERBYDESC** clauses in the URL encoded 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. For more information, see Listing Values in Order.

getKeys

Get the `sys_id` of multiple records by specifying an encoded query string in the `sysparm_query` parameter.

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=getKeys&sysparm_query=active=true^category=hardware
```

get

Get a record directly by specifying the `sys_id` in a `sysparm_sys_id` parameter.

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_sys_id=9d385017c611228701d22104cc95c371
```

Optionally, you may also specify the `sysparm_action` parameter:

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=get&sysparm_sys_id=9d385017c611228701d22104cc95c371
```

getRecords

Get all records by specifying an encoded query string in the `sysparm_query` parameter.

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=getRecords&sysparm_query=active=true^category=hardware
```

Data Modification API

To modify the JSON web service, issue an HTTPS POST request to the platform. The HTTP POST must contain a `sysparm_action` parameter indicating the type of action to be performed, with the incoming JSON object post in the body.



Note: The content-type of the POST should be `application/json`. It cannot be `application/x-www-form-urlencoded` or `<tt?multipart/form-data`.

insert

Create a new record in ServiceNow. The JSON object has to be POSTed as the body (content-type is usually `application/json`, although not enforced). The response from the record creation is a JSON object of the incident that was created.

For example, posting the following JSON object:

```
{"short_description":"this is a test","priority":"1"}
```

to the following URL:

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=insert
```

creates an incident.

Optionally, you may also specify the `sysparm_action` in the JSON object. The parameter inside the JSON object takes precedence over the URL parameter. For example:

```
{"sysparm_action":"insert","short_description":"this is a test","priority":"1"}
```

Inserting Multiple Records

To create multiple new records in ServiceNow, the input JSON object for the `insert` function must be an array. The response from the record creation is a JSON object of the incidents that were created. For example, the following JSON object posted to one the following URLs creates two incidents. Note the fields described as an array value for the **records** field.

```
{"records":[{"short_description":"this was inserted with python using
JSON 1", "priority": "1 - Critical", "impact":"1", "caller_id":"Fred
Luddy"},
            {"short_description":"this was inserted with python using
JSON 2", "priority": "1 - Critical", "impact":"1", "caller_id":"Fred
Luddy"}]}
```

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=insert
```

```
https://<instance name>.service-now.com/incident.do?JSON&sysparm_action=insertMultiple
```

update

Update a record or a list of records filtered by an encoded query string specified by the `sysparm_query` URL parameter. The JSON object has to be posted as the body (content-type is usually `application/json`, although not enforced). The response from the record creation is an array of JSON objects representing the records that were updated.

For example, posting the following JSON object:

```
{"short_description":"this was updated with python", "priority": "3", "impact":"1"}
```

to the following URL:

```
https://instance_name.service-now.com/incident.do?JSON&sysparm_query=priority=3&sysparm_action=update
```

updates all incidents with priority 3, and sets the values specified by the JSON object.

deleteRecord

Delete a single record, identified by a `sysparm_sys_id` parameter, from the targeted table. The parameter may be encoded in the input JSON object or given as a URL parameter.

Posting:

```
{"sysparm_sys_id":"fd4001f80a0a0b380032ffa2b749927b"}
```

to the following URL:

```
http://instance_name.service-now.com/incident.do?JSON&sysparm_action=deleteRecord
```

deletes the incident record identified by the `sys_id` `fd4001f80a0a0b380032ffa2b749927b`.

deleteMultiple

Delete multiple records from the targeted table, filtered by an encoded query string specified in the `sysparm_query` URL parameter. The filter may also be encoded in the input JSON object.

Posting:

```
{"sysparm_query": "short_description=this was updated with python"}
```

to the following URL:

```
http://instance_name.service-now.com/incident.do?JSON&sysparm_action=deleteMultiple
```

deletes all incident records where the **short_description** field contains the value **this was updated with python**.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-other-web-services/concept/c_JSONv2WebService.html
- [2] <http://www.json.org/>

CSV Web Service

CSV



Note: This article applies to Fuji. For more current information, see *CSV 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

The ServiceNow platform supports programmatic retrieval of CSV ^[2] data through an HTTP GET request. The request is triggered by use of a URL parameter **CSV**. For example, to retrieve a list of "incidents" and save to a CSV file, you may issue an HTTP GET to the following URL:

```
https://instance_name.service-now.com/incident.do?CSV
```

The "content-disposition" HTTP header in the response will indicate the file name and extension of the extract. In the case of the example above, it will be "incident.csv"

Parameters

The following URL parameters are supported to customize and filter the response you get.

sysparm_query

The value for this URL parameter is that of an encoded query string and when used, will filter the data using the encoded query before returning the CSV content. The following request will filter the list to only return "incidents" that are active:

```
https://instance_name.service-now.com/incident.do?CSV&sysparm_query=active=true
```

sysparm_view

The value of this URL parameter indicates which list view to use to limit the field values that are returned. For example, to use the "ess" view:

```
https://instance_name.service-now.com/incident.do?CSV&sysparm_view=ess
```

Security

Web service security is enforced using the combination of basic authentication challenge/response for the HTTP protocol, as well as system level access control using the Contextual Security Manager.

To enforce basic authentication on each Web Service request, each request would have to contain the **Authorization** header as specified in the Basic Authentication ^[3] protocol. Because the request is non-interactive, we always require the **Authorization** header during a request.

Supplying basic authentication information whether or not it is required has the added advantage that the data created or updated as a result of the Web Service invocation is done on behalf of the user supplied in the basic authentication credentials. As an example, when creating an Incident record, the journal fields will have the user ID of basic authenticated user, instead of the default "Guest" user.

Posting a CSV file directly

You can also create new data by posting a CSV file directly to a table, the requirement being that the CSV file headers must match the field columns in the targeted table. More detail on this technique can be found [here](#)

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-other-web-services/concept/c_CSVWebService.html
- [2] http://en.wikipedia.org/wiki/Comma-separated_values

EXCEL Web Service

EXCEL



Note: This article applies to Fuji and earlier releases. For more current information, see *Excel Web Service* ^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The ServiceNow platform supports programmatic retrieval of Excel binary formatted ^[2] data through an HTTP GET request. The request is triggered by use of a URL parameter **EXCEL**. For example, to retrieve a list of "incidents" and save to a Excel file, you may issue an HTTP GET to the following URL:

```
https://instance_name.service-now.com/incident.do?EXCEL
```

The "content-disposition" HTTP header in the response will indicate the file name and extension of the extract. In the case of the example above, it will be "incident.xls"

Parameters

The following URL parameters are supported to customize and filter the response you get.

sysparm_query

The value for this URL parameter is that of an encoded query string and when used, will filter the data using the encoded query before returning the Excel file. The following request will filter the list to only return "incidents" that are active:

```
https://instance_name.service-now.com/incident.do?EXCEL&sysparm_query=active=true
```

sysparm_view

The value of this URL parameter indicates which list view to use to limit the field values that are returned. For example, to use the "ess" view:

```
https://instance_name.service-now.com/incident.do?EXCEL&sysparm_view=ess
```

Security

Web service security is enforced using the combination of basic authentication challenge/response for the HTTP protocol, as well as system level access control using the Contextual Security Manager.

To enforce basic authentication on each Web Service request, each request would have to contain the **Authorization** header as specified in the Basic Authentication ^[3] protocol. Because the request is non-interactive, we always require the **Authorization** header during a request.

Supplying basic authentication information whether or not it is required has the added advantage that the data created or updated as a result of the Web Service invocation is done on behalf of the user supplied in the basic authentication credentials. As an example, when creating an Incident record, the journal fields will have the user ID of basic authenticated user, instead of the default "Guest" user.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-other-web-services/concept/c_EXCELWebService.html
 [2] http://en.wikipedia.org/wiki/Microsoft_Excel_file_format#Binary

XML



Note: This article applies to Fuji and earlier releases. For more current information, see *XML Web Service* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The ServiceNow platform supports programmatic retrieval of XML data through a URL query (HTTP GET request). The request is triggered by use of a URL parameter.

Available URL Parameters

ServiceNow supports the following XML-based export formats from a URL query. See Export Format Processors for more information.

Export Processor	URL Syntax	Example URL
Schema	?SCHEMA	https://instance_name.service-now.com/incident.do?SCHEMA
XML	?XML	https://instance_name.service-now.com/incident.do?XML
XSD	?XSD	https://instance_name.service-now.com/incident.do?XSD

The "content-disposition" HTTP header in the response will indicate the file name and extension of the extract. For example, "incident.xml".

Parameters

The following URL parameters are supported to customize and filter the response you get.

sysparm_query

The value for this URL parameter is that of an encoded query string and when used, will filter the data using the encoded query before returning the XML file. The following request will filter the list to only return "incidents" that are active:

```
https://instance_name.service-now.com/incident.do?XML&sysparm_query=active=true
```

useUnloadFormat

This parameter indicates that the XML format that is returned to be of an **unload** format. The **unload** format is the same format you will get if from a list in the UI, you would right-click on a list header and select the Export->XML option. The **unload** formatted XML can be imported back into the instance by administrators of your system. To enable the **unload** format from a URL, use the **useUnloadFormat=true** URL parameter. For example:

```
https://instance_name.service-now.com/incident.do?XML&useUnloadFormat=true
```

Using the URL parameter would produce an XML format that looks like following:

```
<unload unload_date="2010-03-12 17:48:30">
  <incident action="INSERT_OR_UPDATE">
    <description/>
    <category>inquiry</category>
    <due_date/>
    <comments/>
    <knowledge>false</knowledge>
    <business_stc>15261</business_stc>
    <active>false</active>
    <sys_created_on>2010-01-02 22:51:33</sys_created_on>
    <correlation_id/>
    <follow_up/>
    <sys_domain>global</sys_domain>
    <close_notes/>
    <urgency>1</urgency>
    <incident_state>7</incident_state>
    <business_duration>1970-01-01 04:14:21</business_duration>
    <sys_id>46b66a40a9fe198101f243dfbc79033d</sys_id>
    <state>1</state>
    <closed_at>2009-12-28 22:55:16</closed_at>
    <closed_by display_value="Don Goodliffe">9ee1b13dc6112271007f9d0efdb69cd0</closed_by>
    <sys_updated_on>2010-04-12 01:20:57</sys_updated_on>
    <contact_type>phone</contact_type>
    <group_list/>
    <assignment_group display_value=""/>
    <sys_updated_by>glide.maint</sys_updated_by>
    <problem_id display_value="PRB12345">9ee1b13dc6112271007f9d0efd2211d0</problem_id>
    <approval_history/>
```

```

<sys_created_by>don.goodliffe</sys_created_by>
<order/>
<caller_id display_value="Rick Berzle">5137153cc611227c000bbd1bd8cd2006</caller_id>
<variables/>
<calendar_duration>1970-01-02 00:04:53</calendar_duration>
...
</incident>
</unload>

```

Example XML export of a Change Request

The following is an example XML document retrieved using a URL with the following pattern https://instancename.service-now.com/change_request.do?XML. Access control is protected using Basic Authentication [3]

```

<xml>
  <change_request>
    <description>Please install new Cat. 6500 in Data center 01</description>
    <category>Hardware</category>
    <due_date/>
    <scope>3</scope>
    <comments/>
    <knowledge>>false</knowledge>
    <active>true</active>
    <phase>requested</phase>
    <justification/>
    <cab_date/>
    <review_date/>
    <sys_created_on>2009-04-14 23:14:14</sys_created_on>
    <correlation_id/>
    <follow_up/>
    <sys_domain>global</sys_domain>
    <close_notes/>
    <urgency>3</urgency>
    <change_plan/>
    <business_duration/>
    <sys_id>46cb2f54a9fe198101cf6814a2754606</sys_id>
    <state>1</state>
    <reason/>
    <closed_at/>
    <closed_by/>
    <sys_updated_on>2009-12-21 23:47:15</sys_updated_on>
    <contact_type>phone</contact_type>
    <group_list/>
    <risk>3</risk>
    <assignment_group/>
    <sys_updated_by>pat.casey</sys_updated_by>
    <production_system>>false</production_system>
  </change_request>
</xml>

```



```
<approval_history/>
<sys_created_by>glide.maint</sys_created_by>
<end_date>2009-12-21 00:30:00</end_date>
<order/>
<variables/>
<calendar_duration/>
<work_start/>
<backout_plan/>
<start_date>2009-12-20 19:30:00</start_date>
<correlation_display/>
<made_sla>false</made_sla>
<location/>
<test_plan/>
<approval_set/>
<sys_class_name>change_request</sys_class_name>
<sys_mod_count>6</sys_mod_count>
<review_comments/>
<phase_state>open</phase_state>
<time_worked/>
<type/>
<implementation_plan/>
<priority>1</priority>
<user_input/>
<work_end/>
<rejection_goto/>
<upon_reject/>
<requested_by_date>2009-04-08 00:00:00</requested_by_date>
<delivery_plan/>
<parent/>
<delivery_task/>
<short_description>Install new Cisco</short_description>
<assigned_to>681b365ec0a80164000fb0b05854a0cd</assigned_to>
<approval>requested</approval>
<escalation>0</escalation>
<review_status>3</review_status>
<impact>3</impact>
<watch_list/>
<cmdb_ci/>
<cab_recommendation/>
<work_notes/>
<sla_due/>
<company/>
<wf_activity/>
<upon_approval/>
<expected_start/>
<requested_by/>
<opened_at>2009-04-14 23:14:14</opened_at>
```

```
<number>CHG0000008</number>
<opened_by>glide.maint</opened_by>
<activity_due/>
</change_request>
...
</xml>
```

Security

Web service security is enforced using the combination of basic authentication challenge/response for the HTTP protocol, as well as system level access control using the Contextual Security Manager.

To enforce basic authentication on each Web Service request, each request would have to contain the **Authorization** header as specified in the Basic Authentication ^[3] protocol. Because the request is non-interactive, we always require the **Authorization** header during a request.

Supplying basic authentication information whether or not it is required has the added advantage that the data created or updated as a result of the Web Service invocation is done on behalf of the user supplied in the basic authentication credentials. As an example, when creating an Incident record, the journal fields will have the user ID of basic authenticated user, instead of the default "Guest" user.

Record Limits

For information regarding limits on querying records, see Export Limits.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-other-web-services/concept/c_XMLWebService.html

PDF



Note: This article applies to Fuji and earlier releases. For more current information, see *PDF Web Service* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The ServiceNow platform supports programmatic retrieval of PDF ^[2] data through an HTTP GET request. The request is triggered by use of a URL parameter **PDF**. For example, to retrieve a list of "incidents" and save to a PDF file, you may issue an HTTP GET to the following URL:



Note: For PDF export, there is a distinction between targeting a table, and its list. To generate a PDF of a list of records, suffix the target with "_list". To target a single record, you must specify the `sys_id` parameter to identify the record for which you are generating the PDF.

```
https://instance_name.service-now.com/incident_list.do?PDF
```

The "content-disposition" HTTP header in the response will indicate the file name and extension of the extract. In the case of the example above, it will be "incident.pdf"

Parameters

The following URL parameters are supported to customize and filter the response you get.

sysparm_query

The value for this URL parameter is that of an encoded query string and when used, will filter the data using the encoded query before returning the PDF file. The following request will filter the list to only return "incidents" that are active:

```
https://instance_name.service-now.com/incident_list.do?PDF&sysparm_query=active=true
```

Using the sysparm_view Parameter

The value of the `sysparm_view` URL parameter indicates which list view to use to limit the field values that are returned. For example, to use the "ess" view on a single record:

```
https://instance_name.service-now.com/incident.do?PDF&sys_id=0c804d890a0a3c1800e6334c7504a474&sysparm_view=ess
```

Or to generate a list:

```
https://instance_name.service-now.com/incident_list.do?PDF&sysparm_view=ess
```

Security

Web service security is enforced using the combination of basic authentication challenge/response for the HTTP protocol, as well as system level access control using the Contextual Security Manager.

To enforce basic authentication on each Web Service request, each request would have to contain the **Authorization** header as specified in the Basic Authentication ^[3] protocol. Because the request is non-interactive, we always require the **Authorization** header during a request.

Supplying basic authentication information whether or not it is required has the added advantage that the data created or updated as a result of the Web Service invocation is done on behalf of the user supplied in the basic authentication credentials. As an example, when creating an Incident record, the journal fields will have the user ID of basic authenticated user, instead of the default "Guest" user.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/inbound-other-web-services/concept/c_PDFWebService.html
- [2] http://en.wikipedia.org/wiki/Portable_Document_Format

RSS

RSS Feed Generator



Note: This article applies to Fuji and earlier releases. For more current information, see *RSS Feed Generator* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

ServiceNow supports the dynamic generation of RSS ^[2] feeds. Much like our Web Services implementation, the retrieval of an RSS feed representation of information is simply done by specifying an **RSS** parameter at the end of the URL to a table list. For example, the following will return a list of all incidents in RSS 2.0 ^[3] format.:

Adding a Query

To associate a query to the list so that a filtered list is returned, use the **sysparm_query** parameter. For example, the following will return a list of all incidents where the priority field is 1 (Critical):

```
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&RSS
```

If you have a multi part query then you would separate the parts with the ^ character. For example to get all priority 1 incidents with a category of software you would:

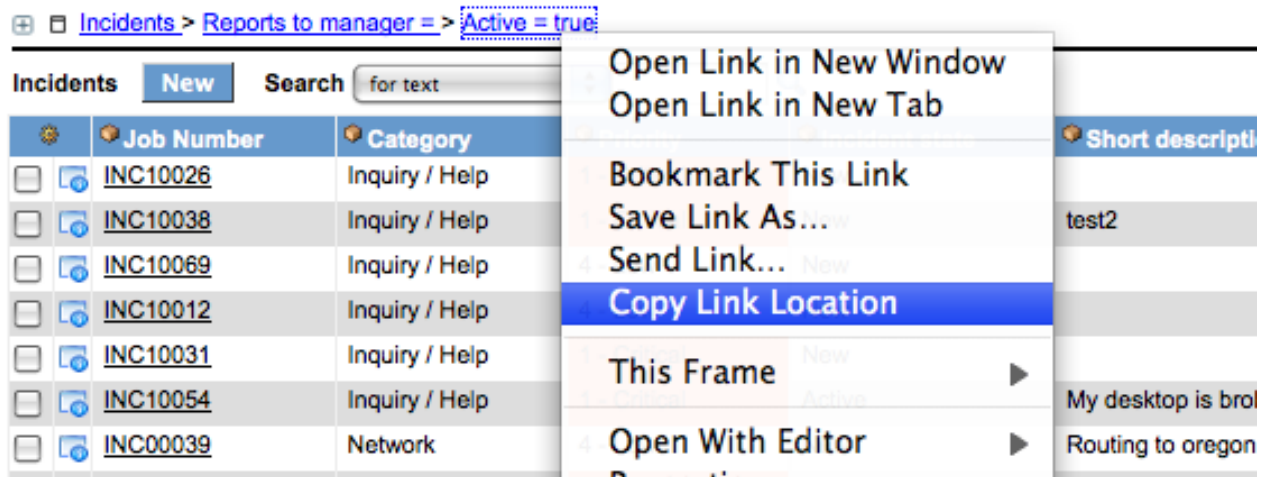
```
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1^category=software&RSS
```

If you want to query on a field that is a reference to another file then you need to use javascript to resolve the reference to the other file. For example, the assigned_to field in incident is a reference to a user record. If you wanted to find all the incidents assigned to "ITIL User" then you would do the following:

```
https://<instance name>.service-now.com/incident.do?sysparm_query=assigned_to=javascript:GetIDValue('sys_user','ITIL%20User')&RSS
```



Note: You can in most cases simply append "&RSS" to a URL that you generate in the U.I. or that of your favorite module. The easiest way to get the URL is to simply click on the last breadcrumb from the list view. After appending "&RSS" then you can use this URL in your RSS feed reader



Limiting results with a view

The description element in the returned RSS xml is constructed using the view as specified in the URL, when no view is specified, the default no-name view is used. To change this format, specify the **sysparm_view** parameter on the URL. For example, the following will return the previous list, however the description element will now be restricted to the fields only available to the **ess** view:

```
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&sysparm_view=ess&RSS
```

Additionally, the RSS item title can be modified using the **sysparm_title_view** URL parameter. When specified, the item title will be constructed using the fields specified in the view. For example:

```
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&sysparm_view=ess&sysparm_title_view=rss_title&RSS
```

Formatting results

The description element in the returned RSS xml can be *formatted* by setting the URL parameter **sysparm_format=true** and specifying the format string in the property *glide.rss.description_format*. By default, when the URL parameter is present, the description element will be formatted to contain the field label and value using the following format string:

```
<b>{1}</b>: {2}<br/>
```

- {0} - field name
- {1} - field label
- {2} - field value

This default format string can be overridden using the property *glide.rss.description_format*. An example of the formatted RSS feed can be seen in the following screen capture from Firefox:

Problem (All)

Problem (All)

[PRB00001](#)

Sun, Aug 10, 2008 3:39 PM

Number: PRB00001

Escalation: Moderate

Short description: Windows xp SP2 causing errors in Enterprise

Problem state: Pending Change

RFC: CHG00003

[PRB00008](#)

Sun, Aug 10, 2008 3:40 PM

Number: PRB00008

Escalation: Normal

Short description: Hang when trying to print VISIO document

Problem state: Open

RFC:

Basic Authentication

To enforce basic authentication on each request for an RSS feed, you may set the property **glide.basicauth.required.rss** to **true**. RSS request would have to contain the "Authorization" header as specified in the Basic Authentication ^[3] protocol. Because the request is non-interactive, we always require the **Authorization** header during a request.



Note: *If you plan to disable RSS basic authentication, make sure that tables in the platform have the right ACL entries to protect from unauthorized access*

In Firefox 2.0, to specify basic authentication on the URL, put the username/password pair separated by ":" in front of the server name before an "@" character, for example, to use the demo credentials for itil to read the feed on our demo server:

```
https://itil:itil@<instance name>.service-now.com/incident.do?RSS
```

Microsoft IE 7 does not support direct URL authentication <http://support.microsoft.com/kb/834489> . If the Web site uses the basic authentication method, Internet Explorer automatically prompts users for a user name and a password. In some cases, users can click the Remember my password box in the dialog box to save their credentials for later visits to that Web site.

Overriding the RSS title

You may also optionally override the automatically generated "title" of the RSS feed by introducing `sysparm_title` parameter to the request URL. For example:

```
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&sysparm_view=ess&RSS&sysparm_title=Priority%20One%20Incidents
```

This will produce results as follows:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <rss version="2.0">
- <channel>
  <title>Priority One Incidents</title>
  <link>http://www.service-now.com/demo/nav_to.do?uri=incident.do?sysparm_query=priority=1</link>
  <description>Priority One Incidents</description>
  <copyright>Copyright 2006 Service-now.com</copyright>
  <pubDate>Mon, 12 Jun 2006 11:04:44 PDT</pubDate>
  <lastBuildDate>Mon, 12 Jun 2006 11:04:44 PDT</lastBuildDate>
  <generator>jRSSGenerator by Henrique A. Viegas</generator>
  <docs>http://blogs.law.harvard.edu/tech/rss</docs>
- <item>
  <title>INC000009</title>
  <link>http://www.service-now.com/demo/nav_to.do?uri=incident.do?
    sys_id=46b66a40a9fe198101f243dfbc79033d%26sysparm_stack=incident_list.do%
    3Fsysparm_query=active=true</link>
  <description>INC000009 2006-02-01 14:50:23 Reset my password</description>
  <author>glide.maint</author>
  <guid>46b66a40a9fe198101f243dfbc79033d</guid>
  <pubDate>Wed, 17 May 2006 18:20:57 PDT</pubDate>
</item>
+ <item>
+ <item>
+ <item>
+ <item>
+ <item>
</channel>
</rss>
```

RSS Feed Readers

To render RSS data properly, one must use an RSS reader. There are RSS readers available in IE7 and Outlook 2007. Here is the IE7 link for reading RSS <http://www.microsoft.com/windows/IE/ie7/tour/rss/>.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/rss/concept/c_RSSFeedGenerator.html
- [2] http://en.wikipedia.org/wiki/RSS_%28file_format%29
- [3] <http://www.rssboard.org/rss-specification>

RSS Feed Reader

Overview

A *scroller* can be created out of anything using a UI page as detailed on the wiki.

http://wiki.service-now.com/index.php?title=Adding_a_Scrolling_News_Panel

Once you have that, you just need to get the HTML content for your UI page and put it in the scroller. There are a few RSS Feed parsers to choose from.

See the following ServiceNow Community post for more information and discussion about using this feature: <https://community.servicenow.com/message/806535> ^[1].

References

[1] <http://community.servicenow.com/message/806535>

ODBC Driver

ODBC Driver



Note: This article applies to Fuji and earlier releases. For more current information, see *ODBC Driver* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The ServiceNow ODBC driver ^[2] allows an ODBC client to connect to the ServiceNow platform for reporting. The driver is compliant to version 3.52 of the Microsoft ODBC core API conformance. The ServiceNow ODBC driver uses the ServiceNow web services support for a *query-only* interface. Because it uses the web services interface, platform-wide access control (ACL) is enforced and data security is in place.



Note: The ServiceNow ODBC driver has these limitations:

- The ODBC driver supports only **SELECT** statements or read-only functions, and does not modify your instance data.
- There is no supported way to use the ODBC driver with a Java client application or with a Java JDBC-ODBC bridge.

Downloading the ODBC Driver

The ODBC driver is available from the ServiceNow Knowledge Base ^[3]. If you do not have access to the Knowledge Base, contact your ServiceNow administrator.



Note: Versions older than 1.0.7.3 of the ODBC Driver are no longer supported.

Working with ODBC

To quickly set up and run the ODBC Driver, use the ODBC quick start guide and video tutorial. Use these links for more detailed setup and configuration instructions:

1. Install the ODBC driver or upgrade an existing ODBC installation.
2. Configure the ODBC driver.
3. Test the configuration.
4. Use your applications with ODBC.
5. Use the ODBC driver, following ODBC best practices.

For troubleshooting information, see the knowledge base articles *troubleshooting ODBC driver issues* ^[4] and *troubleshooting common ODBC error messages* ^[5].

Using Client Applications with the ODBC Driver

See the following pages for examples of how to use the ODBC driver to create data sources from other applications.

- Using Interactive SQL (ODBC)
- Using ODBC Driver in SQL Server
- Using the ODBC Driver in Excel
- Using the ODBC Driver in Crystal Reports

ODBC Troubleshooting Resources

Several ODBC troubleshooting articles are available on the HI knowledge base.

- Troubleshooting ODBC Driver ^[4]
- Troubleshooting common ODBC error messages ^[5]
- Troubleshooting Linked Server ^[6]

Enhancements

Fuji

- You can set the query mode property to use the AND operator.
- The ODBC Driver provides improved logging and allows you to specify the log file location and logging level using a JVM option.

Dublin

- When export query strings become large enough to impact performance, the ODBC driver converts the data type from VARCHAR to the LONGVARCHAR data type.
- The ODBC driver respects character limits set in the dictionary.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/odbc-driver/concept/c_ODBCDriver.html
 - [2] https://docs.servicenow.com/bundle/helsinki-servicenow-platform/page/integrate/odbc-driver/concept/c_ODBCDriver.html
 - [3] https://hi.service-now.com/kb_view.do?sysparm_article=KB0540707
 - [4] https://hi.service-now.com/kb_view.do?sysparm_article=KB0538943
 - [5] https://hi.service-now.com/kb_view.do?sysparm_article=KB0538954
 - [6] https://hi.service-now.com/kb_view.do?sysparm_article=KB0538992
-

HTTP Client Connection Management

Overview

Outbound HTTP(S) connections from a ServiceNow instance or inbound connections from MID Servers, the ODBC driver, and other clients are maintained and reused where possible. Connection pooling is used to keep track of HTTP(S) client connections to determine if they are alive and available for reuse.

ServiceNow HTTP *client code* means:

- Any application or script which makes outbound HTTP(S) requests from a ServiceNow instance.
- ServiceNow code in the MID Server or the ODBC driver which makes HTTP(S) requests to one or more ServiceNow instances.



Note: This discussion does not apply to browser-to-instance communication. No changes have been made with respect to the management of HTTP(S) connections for browser-based communication with ServiceNow. This discussion also does not apply to customer-developed Web Services clients making requests to ServiceNow.

HTTP Connection Management Properties

Connection pooling is controlled by three properties. The default values for these properties are appropriate for most customers. The Glide properties are dynamic, meaning that changes to these properties will take effect immediately. No outage or restart is required to update the values.

Property	Description	Default Value
<code>glide.http.use_connection_mgr</code>	Switches connection pooling on and off. To disable the new behavior (not recommended) set <code>glide.http.use_connection_mgr</code> to <i>false</i> .	true
<code>glide.http.connection_mgr.max_connections</code>	Controls the total number of permitted HTTP(S) connections outbound from the ServiceNow instance. This is an instance-wide value.	20
<code>glide.http.connection_mgr.max_connections_per_host</code>	Controls how many of the <code>glide.http.connection_mgr.max_connections</code> can communicate in parallel with any particular host. If the maximum setting for any of these values is reached during normal operations, a script or background thread may have to wait briefly to obtain a connection.	4

What Should the Customer Do?

Users should monitor performance, such as the decreased time for loading Discovery data and improved ODBC driver performance. For systems with an unusually large amount of simultaneous outbound HTTP(S) activity, such as numerous third-party integrations or high-volume automated activities which generate HTTP(S) requests from the ServiceNow instance to other places, review the **max_connections** and **max_connections_per_host** properties to ensure that the settings are sufficient. This enhancement has no impact on end-user connections from browsers and no impact on connections from customer-developed Web Services client applications.

Running Interactive SQL (ODBC)



Note: This article applies to Fuji and earlier releases. For more current information, see *Using Interactive SQL with ODBC*^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

Run the interactive SQL application for quick verification of connectivity and to test query results without using a full application.

Configuration

1. In Windows, navigate to **Start > Programs > ServiceNow ODBC > Interactive SQL (ODBC)**.
2. Enter the following command to connect to the ServiceNow instance. Select the appropriate user credentials in the format: `ID*password@DSNName`. The password cannot contain special characters.

```
connect odbcuser*password@ServiceNow
```

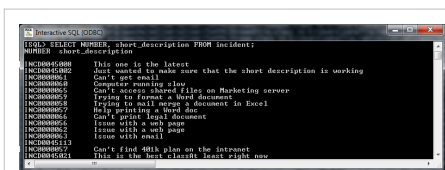


Connecting the SQL

3. Issue a **SELECT** SQL command, such as:

```
select number, short_description from incident;
```

Make sure to include the semicolon at the end of your query statement. You will be presented with a 'Cont>' prompt otherwise.



Sample SQL query

SQL Support

The ODBC driver embeds a third party SQL/ODBC engine from DataDirect, a division of Progress Software. See the *DataDirect SQL Reference*^[2] for information on proper SQL syntax.



Note: The ServiceNow ODBC driver only supports *SELECT* statements. The driver ignores other SQL statements such as *CREATE* and *ALTER*.

Specifying the Maximum Number of Rows Returned

By default, ServiceNow only returns 100 rows of data with each iSQL query. If you need to return more rows of data, set the `maxrows` parameter for the iSQL session.

To return all rows set `maxrows` to zero:

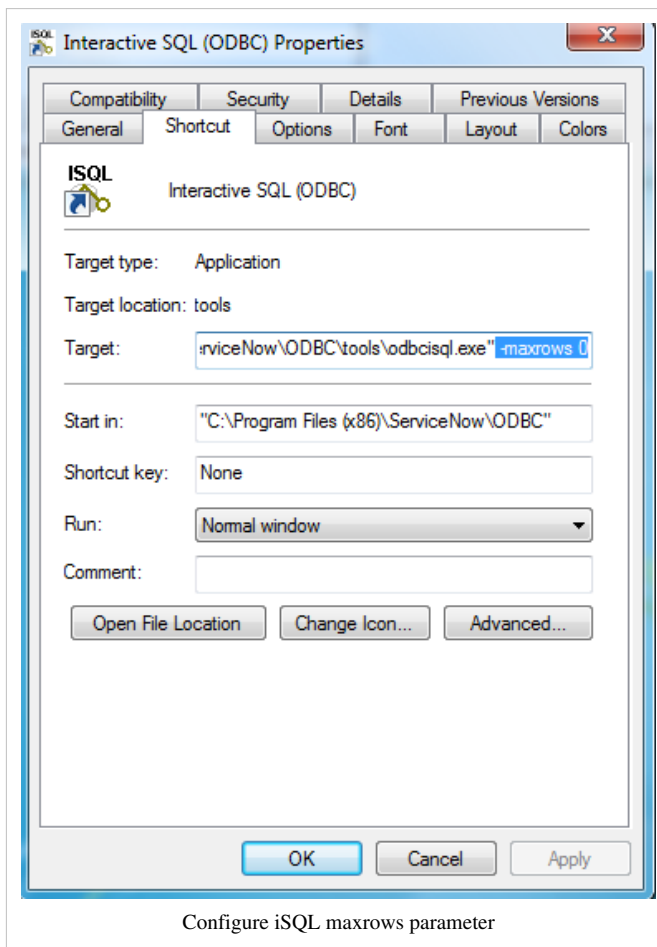
```
maxrows 0
```

To return more than 100 rows set `maxrows` to a higher value. For example, to return 500 rows:

```
maxrows 500
```



Note: If running the Interactive SQL console from a shortcut, you must modify the shortcut **Target** to include the **-maxrows** parameter with the desired value.



References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/odbc-driver/task/t_UsingInteractiveSQLWithODBC.html
- [2] <http://media.datadirect.com/download/docs/openaccess/alloy/sqlref/wwhelp/wwhimpl/js/html/wwhelp.htm#href=preface.2.3.html>

Using the ODBC Driver in Excel 2010



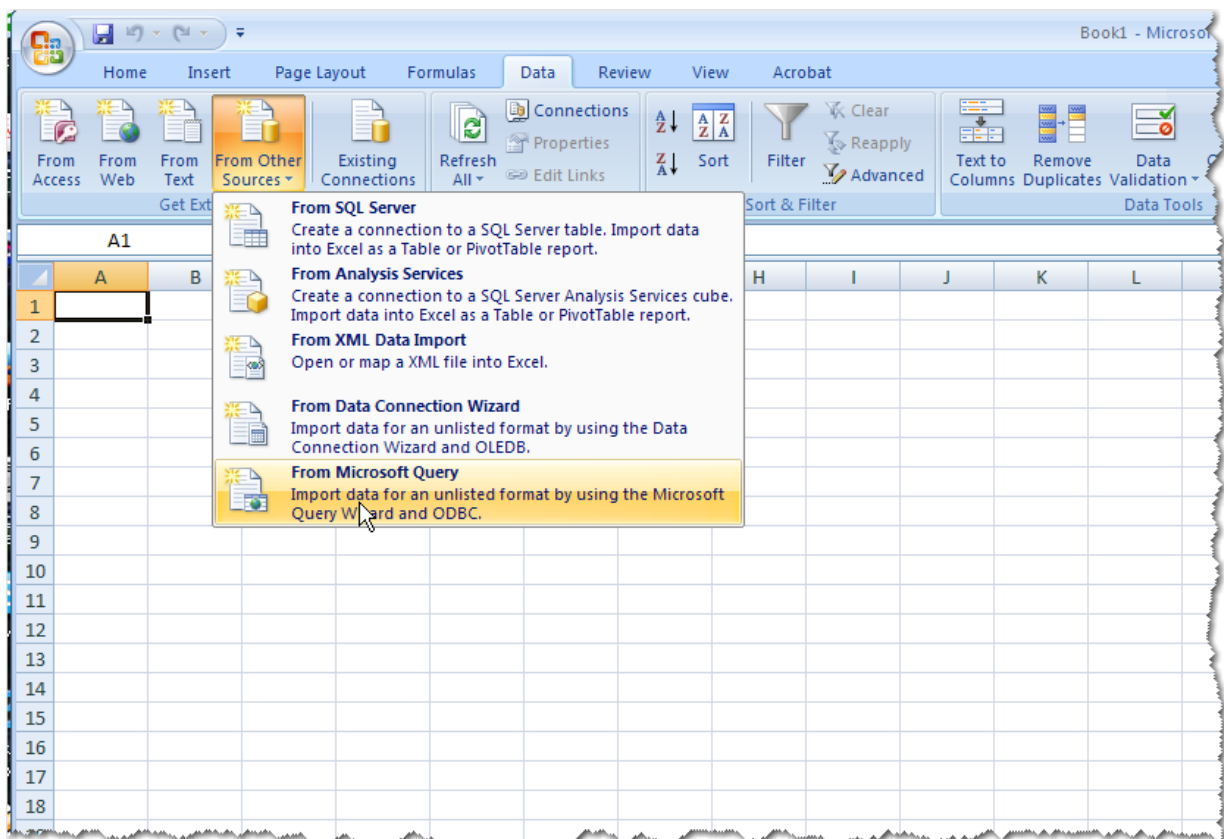
Note: This article applies to Fuji and earlier releases. For more current information, see *Use the ODBC Driver in Excel* ^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

After installing the ODBC driver and its associated DSN, use it in Excel as a data source provider.

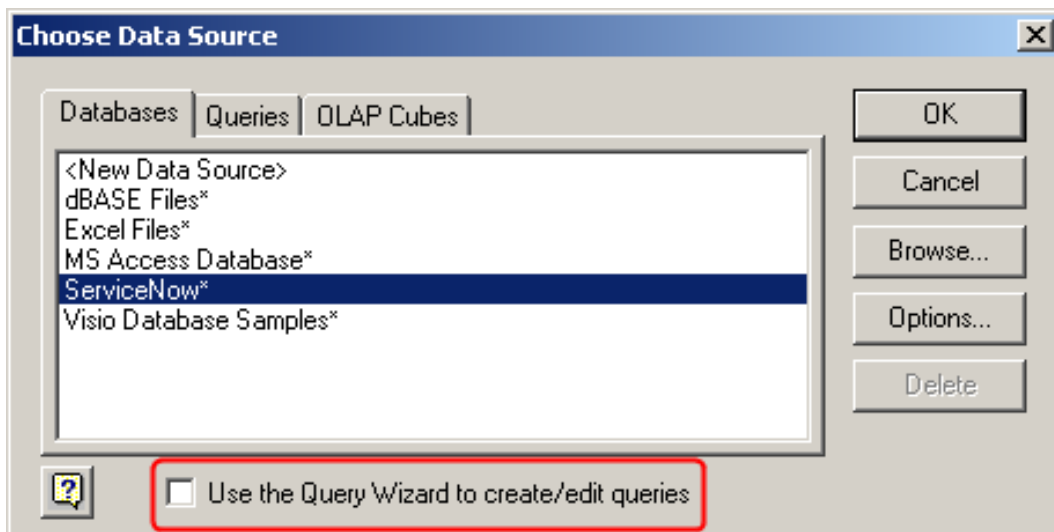
Configuration

1. In Excel open the Data tab.
2. Under **From Other Sources** open **From Microsoft Query**.

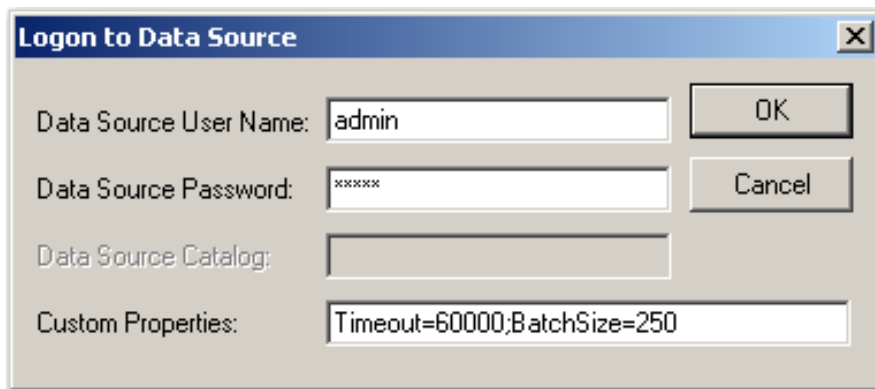


3. Select **ServiceNow** as your database (the default DSN name).
4. Clear the **Use the Query Wizard to create/edit queries** check box.

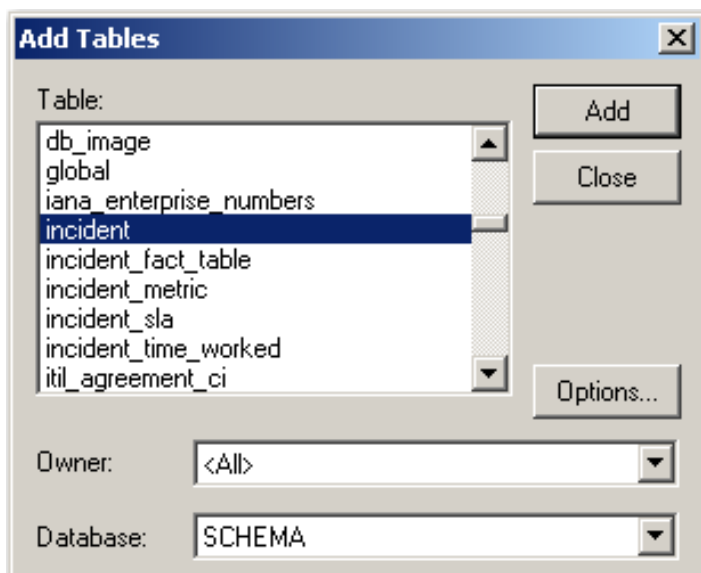
Note: The Excel Query Wizard does not support the listing of columns from a table name that contain an underscore (_). Clearing this check box uses the Query Builder instead, which supports the use of this character.



5. Supply the ServiceNow user name and password that was pre-configured during the ODBC driver configuration procedure.



6. Select a table from the ServiceNow instance and click **Add**.



7. Close the dialog box.
8. Select the table columns from which the Query Builder will retrieve data. Use the list above the table, or type the names directly into the columns, and then press **Enter**.
9. To retrieve the data and create the Excel record, click the Return Data icon or select **File > Return Data to Microsoft Office Excel**.

Microsoft Query

File Edit View Format Table Criteria Records Window Help

Query from local

incident

rejection_goto
rfc
severity
short_description
sla_due
state
subcategory
sys_class_name

number	short_description	dv_assigned_to	dv_state	state
INC0000009	Reset my password	David Loo	Open	1
INC0000010	Need Oracle 10GR2 installed	Don Goodliffe	Open	1
INC0000011	Need new Blackberry setup	ITIL User	Open	1
INC0000012	eFax is not working	David Loo	Open	1
INC0000013	EMAIL is slow when an attach	David Loo	Open	1
INC0000014	missing my home directory	ITIL User	Open	1
INC0000015	I can't launch my game anymo	Don Goodliffe	Open	1
INC0000016	Rain is leaking on main DNS S	ITIL User	Open	1
INC0000017	How do I create a sub-folder	David Loo	Open	1
INC0000018	Sales forecast spreadsheet is I	ITIL User	Open	1
INC0000019	Can't launch X-Win32	Bud Richman	Open	1
INC0000020	Request for a Blackberry	ITIL User	Open	1
INC0000021	New employee hire	Beth Anglin	Open	1
INC0000024	Issue with a web page	ITIL User	Open	1
INC0000025	I need more memory	ITIL User	Open	1
INC0000026	Seem to have an issue with my	Don Goodliffe	Open	1
INC0000028	My disk is still having issues. C	Don Goodliffe	Open	1
INC0000027	please remove this hotfix	ITIL User	Open	1
INC0000029	I cant get my weather report	Don Goodliffe	Open	1
INC0000030	Lost connection to the wireles	David Loo	Open	1
INC0000031	EMAIL Server Down	David Loo	Open	1
INC0000032	EMAIL Server Down Again	David Loo	Open	1
INC0000033	File Server is 80% full - Needs	Don Goodliffe	Open	1
INC0000034	Does not look like a backup o	David Loo	Open	1
INC0000035	Reset my password	Luke Wilson	Open	1
INC0000036	Issue with networking	Luke Wilson	Open	1
INC0000037	Request for a new service	Howard Johnson	Open	1

The requested data is brought into Excel.

Table Name: Table_Query_fro

Table Tools

Home Insert Page Layout Formulas Data Review View

Design

Table Name: Table_Query_fro

Table Style Options

Header Row First Column

Total Row Last Column

Banded Rows Banded Column

number	short_description	dv_assigned_to	dv_state	state
INC0000009	Reset my password	David Loo	Open	1
INC0000010	Need Oracle 10GR2 installed	Don Goodliffe	Open	1
INC0000011	Need new Blackberry setup	ITIL User	Open	1
INC0000012	eFax is not working	David Loo	Open	1
INC0000013	EMAIL is slow when an attachment isinvolved	David Loo	Open	1
INC0000014	missing my home directory	ITIL User	Open	1
INC0000015	I can't launch my game anymore	Don Goodliffe	Open	1
INC0000016	Rain is leaking on main DNS Server	ITIL User	Open	1
INC0000017	How do I create a sub-folder	David Loo	Open	1
INC0000018	Sales forecast spreadsheet is READ ONLY	ITIL User	Open	1
INC0000019	Can't launch X-Win32	Bud Richman	Open	1
INC0000020	Request for a Blackberry	ITIL User	Open	1
INC0000021	New employee hire	Beth Anglin	Open	1
INC0000024	Issue with a web page	ITIL User	Open	1
INC0000025	I need more memory	ITIL User	Open	1
INC0000026	Seem to have an issue with my harddrive...	Don Goodliffe	Open	1
INC0000028	My disk is still having issues. Can't delete afile	Don Goodliffe	Open	1
INC0000027	please remove this hotfix	ITIL User	Open	1
INC0000029	I cant get my weather report	Don Goodliffe	Open	1
INC0000030	Lost connection to the wirelessnetwork	David Loo	Open	1
INC0000031	EMAIL Server Down	David Loo	Open	1
INC0000032	EMAIL Server Down Again	David Loo	Open	1

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/odbc-driver/task/t_UsingTheODBCDriverInExcel2010.html

Using the ODBC Driver in Crystal Reports 2008



Note: This article applies to Fuji and earlier releases. For more current information, see *Use the ODBC Driver in Crystal Reports* [1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

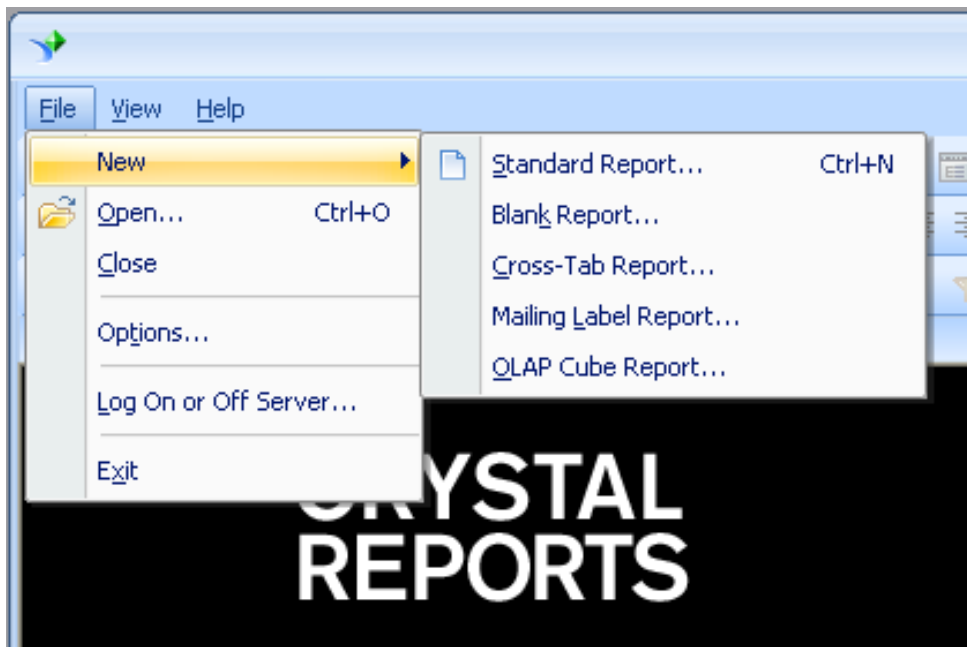
After installing the ODBC driver and its associated DSN, use it in Crystal Reports as a data source provider.

JVM Heap Size Requirements

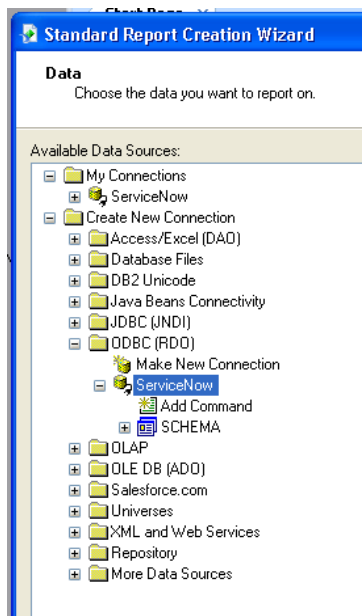
Crystal Reports includes the configuration file *CRConfig.xml* that contains the JVM minimum heap size (Xms) and maximum heap size (Xmx) values. When configuring the ODBC Driver with Crystal Reports, ensure the ODBC Driver uses the same minimum and maximum JVM heap size as Crystal Reports. If these values do not match, update the ODBC Driver settings, not the Crystal Reports settings.

Configuring Crystal Reports

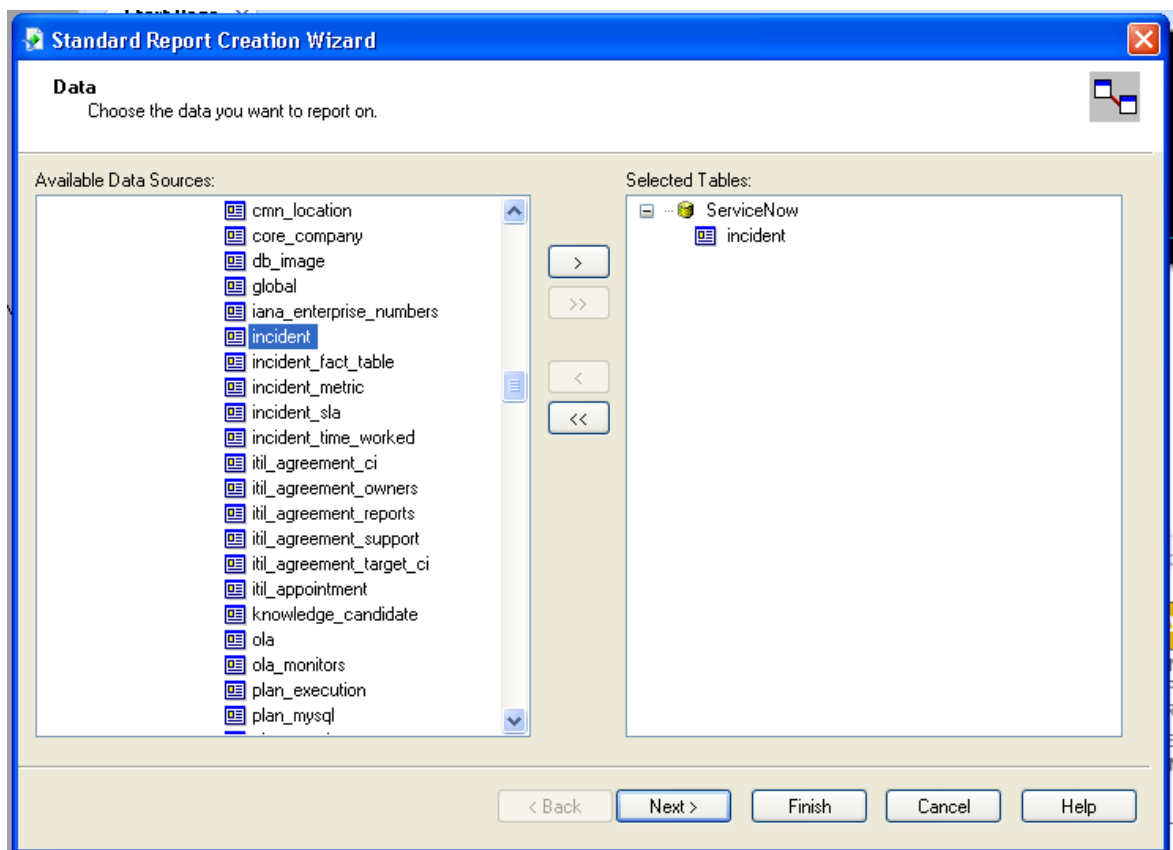
1. Create a new Standard Report



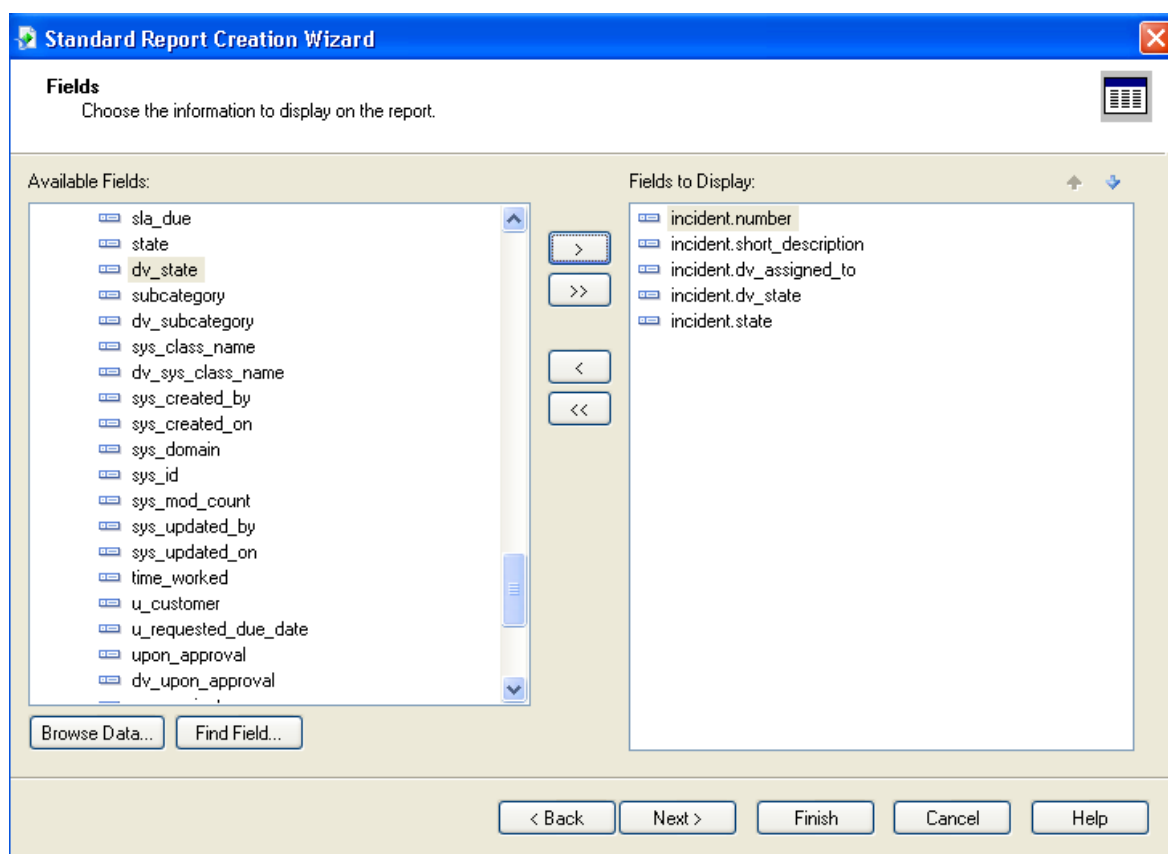
2. Create a new connection using the ServiceNow DSN



3. Select a table from the list of available tables



4. Select the available fields from the selected table



5. Click "Finish" to render the report

Crystal Reports - [Report2]

File Edit View Insert Format Database Report Window Help

1 of 1+

Start Page Report2 x

Design Preview x

Groups

Report2

	number	short_description	dv_assigned_to	dv_state	state
D	INC000009	Reset my password	David Loo	Open	1
D	INC000010	Need Oracle 10GR2 installer	Don Goodliffe	Open	1
D	INC000011	Need new Blackberry setup	ITIL User	Open	1
D	INC000012	eFax is not working	David Loo	Open	1
D	INC000013	EMAIL is slow when an attac	David Loo	Open	1
D	INC000014	missing my home directory		Open	1
D	INC000015	I can't launch my game anyrr		Open	1
D	INC000016	Rain is leaking on main DNS		Open	1
D	INC000017	How do I create a sub-folder		Open	1
D	INC000018	Sales forecast spreadsheet i		Open	1
D	INC000019	Can't launch X-Win32		Open	1
D	INC000020	Request for a Blackberry		Open	1
D	INC000021	New employee hire	Beth Anglin	Open	1
D	INC000024	Issue with a web page	ITIL User	Open	1
D	INC000025	I need more memory		Open	1
D	INC000026	Seem to have an issue with r	Don Goodliffe	Open	1
D	INC000028	My disk is still having issues.	Don Goodliffe	Open	1
D	INC000027	please remove this hotfix	System Administrator	Open	1
D	INC000029	I cant get my weather report		Open	1
D	INC000030	Lost connection to the wirele	David Loo	Open	1
D	INC000031	EMAIL Server Down		Open	1
D	INC000032	EMAIL Server Down Again	David Loo	Open	1

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/odbc-driver/task/t_UseODBCDriverCrysRpts.html

Using ODBC Driver in SQL Server 2008



Note: This article applies to Fuji and earlier releases. For more current information, see *OCBC Driver in SQL Server* ^[1] at <http://docs.servicenow.com> **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

Use the ServiceNow ODBC driver in SQL Server as a *Linked Server* ^[2]. This allows SQL Server to query tables from a ServiceNow instance directly via the ODBC Driver. Use these procedures only with a supported version of SQL Server. Other versions of SQL Server may cause unexpected behavior. If you encounter unexpected behavior, refer to the troubleshooting linked server ^[6] knowledge article.

Video Tutorials

[Configure Microsoft SQL Linked Server](#) [Troubleshooting Linked Server Permissions](#)

Required Permissions

Additional information on the required permissions for SQL Server Linked Servers can be found on the MSDN blog ^[3].

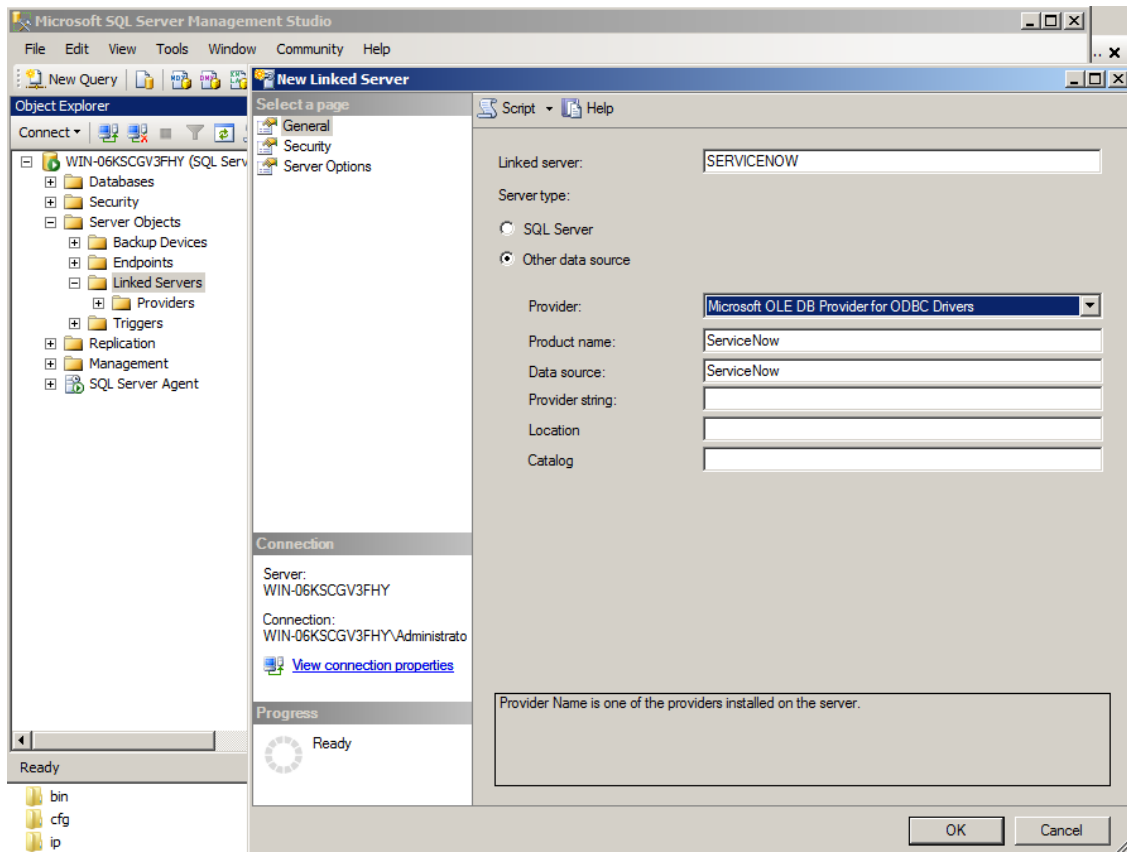


Note: Review this information if you encounter permission errors with SQL Server.

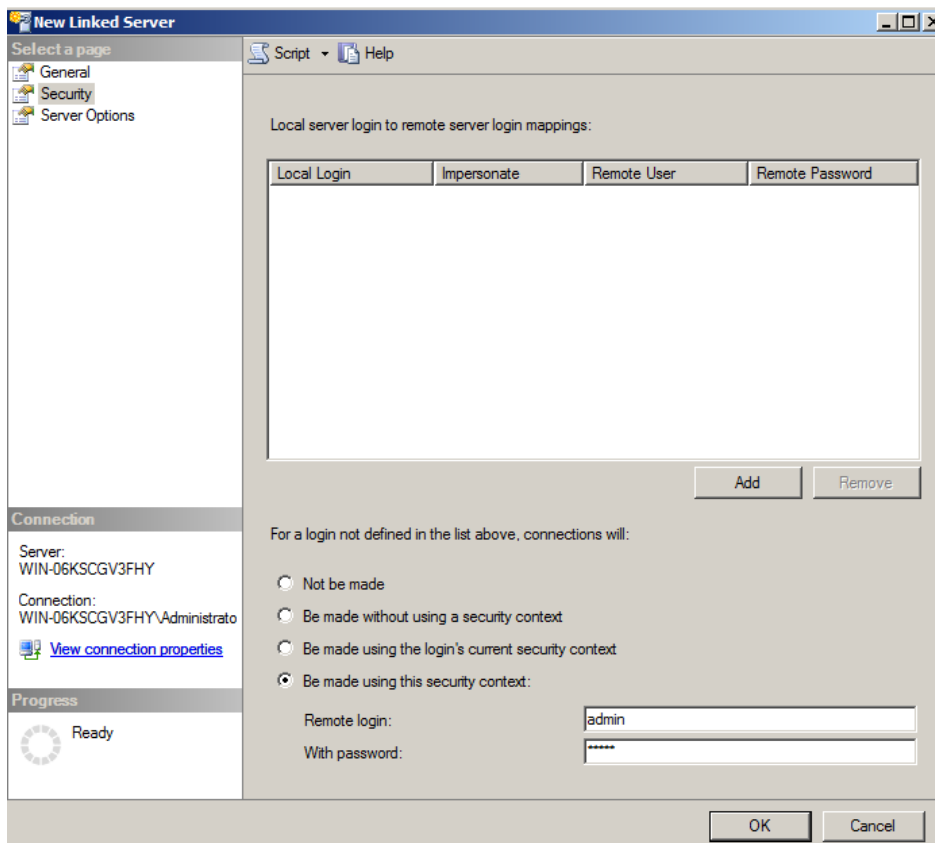
Configuring SQL Server

The following example configuration was performed on SQL Server 2008, installed on Windows Server 2008.

1. Right-click the SQL Server Management Studio application and select as **Run as Administrator**.
2. Log in to the database to which you want to link.
3. Right-click on **Server Objects > Linked Servers**.
4. Click **New Linked Server**.
5. Enter the following values in the dialog.
 - Linked server: **SERVICENOW**. This is the name of the Linked Server.
 - Provider: **Microsoft OLE DB Provider for ODBC Drivers**
 - Product name: **ServiceNow**. This is an identifier. Enter any value that is appropriate.
 - Data source: **ServiceNow**. This is the name of your DSN.



6. Select **Security** from the **Select a page** list, and then enter the following security values:
 - a. For a login connection, select **Be made using this security context**.
 - b. Enter the user name and password for connecting to the ServiceNow instance.
 - c. Click **OK**.

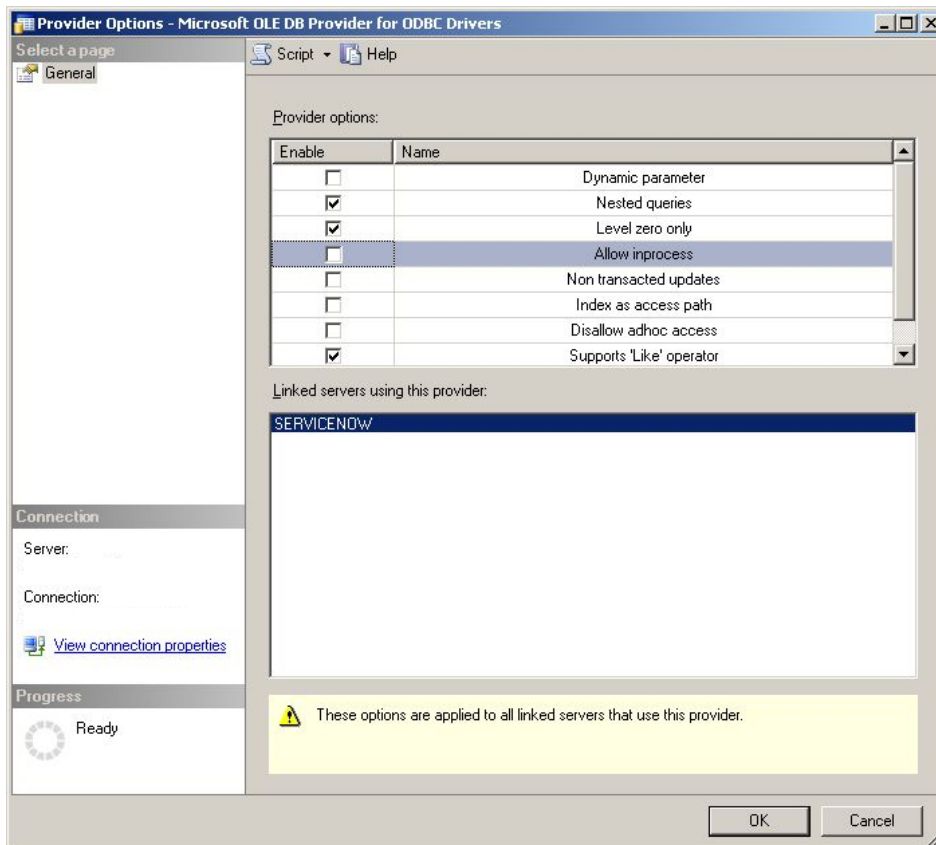


7. Navigate to **Server Objects > Linked Server > Providers** and double-click **Microsoft OLE DB Provider for ODBC Drivers**.

8. Select the following options.

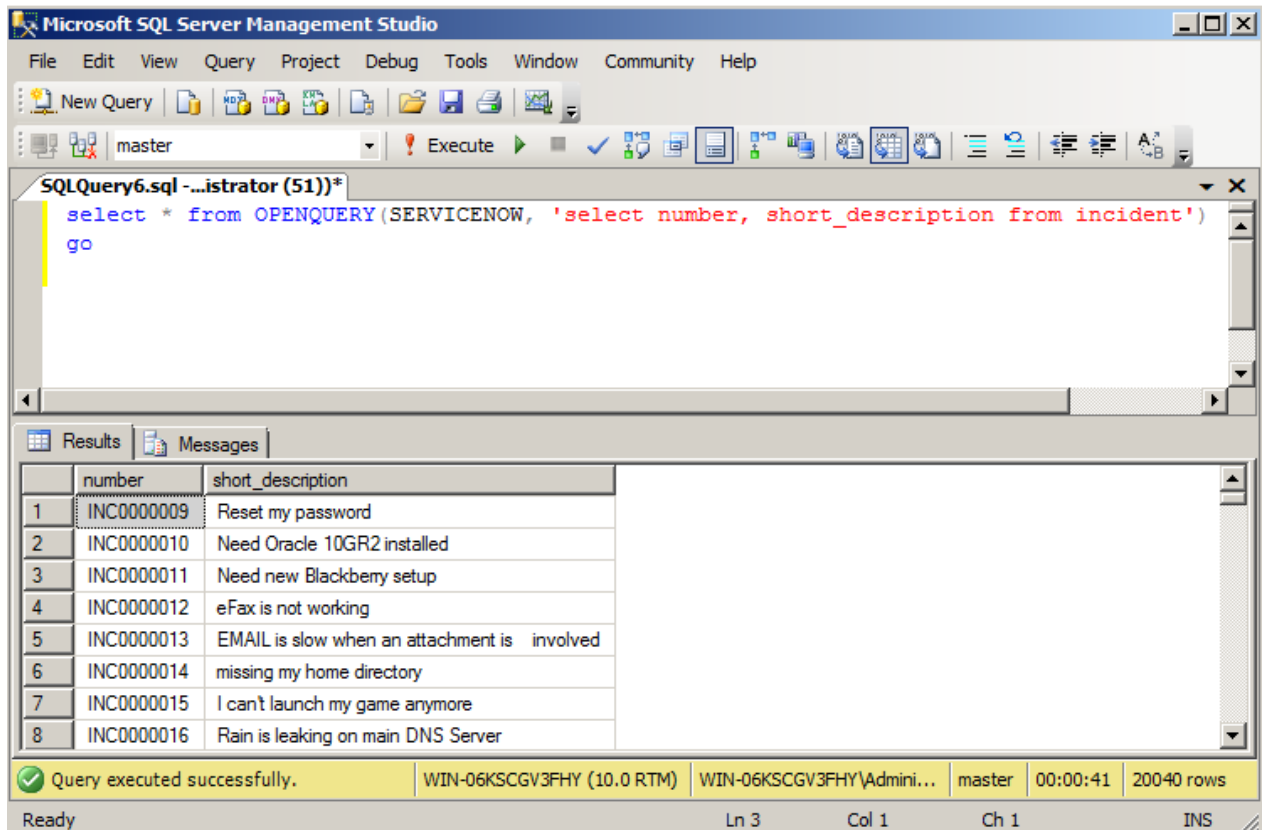
- Nested Queries
- Level zero only
- Support 'Like' operator

Tip: ServiceNow recommends running the third-party provider in the out-of-process mode setting (AllowInProcess=FALSE). If you run the provider in-process (within the same process as SQL Server), then any issues with the provider can affect the SQL Server process, which in turn could result in crashing SQL server.



9. Test your connection by selecting the newly created linked server **SERVICENOW** and selecting **Test connection**.

10. Execute the following query in a query builder window to retrieve some results.



SQL Server Connection String

To use the ODBC driver directly in SQL Server 2008, specify the connection string in the following format.

```
Dsn=ServiceNow;uid=username;pwd=password
```



Note: The latest SQL Server 2008 patches are required for the ability to specify a connection string in the user interface, via the SQL import wizard

Using sp_addlinkedserver

The following example creates a linked server named "ServiceNow ODBC" that uses the Microsoft OLE DB Provider for ODBC (MSDASQL) and the data_source parameter

```
EXEC sp_addlinkedserver
    @server = N'ServiceNow ODBC',
    @srvproduct = N'',
    @provider = N'MSDASQL',
    @datasrc = N'ServiceNow';
GO
```

After creating the linked server, you must update its properties to specify the login credentials.

Troubleshooting Linked Server

You may encounter an error when configuring a linked server with the ODBC Driver. Solutions to common errors are available.

Comprehensive troubleshooting resources are available in the HI knowledge base ^[4].

Access Denied Errors

You may encounter a permissions error when running a query.

```
Msg 7399, Level 16, State 1, Line 1
The OLE DB provider "MSDASQL" for linked server "SERVICENOW" reported an error. Access denied.
Msg 7350, Level 16, State 2, Line 1
Cannot get the column information from OLE DB provider "MSDASQL" for linked server "SERVICENOW".
```

To resolve this error, configured the linked server to run with an out-of-process provider ^[3].

Number Precision Errors

You may encounter precision errors querying for decimal or number field values using the **OPENQUERY** syntax with the ODBC driver. In this case, use the **Cast** syntax to convert the precision. For example:

```
select * from OPENQUERY (SERVICENOW, 'select Cast(sys_mod_count as
Decimal(38,0)), number, short_description from incident')
go
```

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/integrate/odbc-driver/concept/c_ODBCDrvSQL20082012.html
- [2] <http://msdn.microsoft.com/en-us/library/ms188279.aspx>
- [3] <http://blogs.msdn.com/b/dataaccesstechnologies/archive/2010/08/19/permissions-needed-to-set-up-linked-server-with-out-of-process-provider.aspx>
- [4] https://hi.service-now.com/kb_view_customer.do?sysparm_article=KB0538992

Article Sources and Contributors

Web Services *Source:* <http://wiki.servicenow.com/index.php?oldid=251052> *Contributors:* David Loo, Fuji.publishing.user, Joe.Westrich, Joe.zucker, John.ramos, Joseph.messerschmidt, Mary.stromberg, Peter.smith, Rachel.sienko, Steven.wood

SOAP *Source:* <http://wiki.servicenow.com/index.php?oldid=250513> *Contributors:* David Loo, Fuji.publishing.user, G.yedwab, Guy.yedwab, Jeremy.norris, John.andersen, John.ramos, Joseph.messerschmidt, Michelle.Corona, Rachel.sienko, Steven.wood, Vaughn.romero, Wallymarx

Direct Web Services *Source:* <http://wiki.servicenow.com/index.php?oldid=250507> *Contributors:* Ajandersen, CapaJC, Christen.mitchell, David Loo, Dkearney, Fred.luddy, Fuji.publishing.user, G.yedwab, Gflewis, Guy.yedwab, Jessi.graves, Jim.uebbing, John.ramos, Jonathan.sparks, Joseph.messerschmidt, Julie.phaviseth, Mark.stanger, Martin.wood, Rachel.sienko, Ray.LeBlanc, Rob.woodbyrne, Steven.wood, Valor, Vaughn.romero, Vhearne, Wallymarx

Direct SOAP API *Source:* <http://wiki.servicenow.com/index.php?oldid=240556> *Contributors:* Fuji.publishing.user, John.ramos, Joseph.messerschmidt

Web Service Import Sets *Source:* <http://wiki.servicenow.com/index.php?oldid=251051> *Contributors:* CapaJC, David Loo, Fuji.publishing.user, G.yedwab, George.rawlins, Guy.yedwab, Jim.uebbing, Joe.Westrich, John.andersen, John.ramos, Joseph.messerschmidt, Rachel.sienko, Steven.wood, Vaughn.romero, Vhearne

Web Service Consumer *Source:* <http://wiki.servicenow.com/index.php?oldid=242090> *Contributors:* Fuji.publishing.user, John.ramos, Joseph.messerschmidt, Mary.stromberg, Steven.wood

Scripted Web Services *Source:* <http://wiki.servicenow.com/index.php?oldid=250850> *Contributors:* Ajandersen, Anthony.devine, David Loo, Emily.partridge, Fuji.publishing.user, Guy.yedwab, John.ramos, Joseph.messerschmidt, Rachel.sienko, Steven.wood

Hierarchical Web Service *Source:* <http://wiki.servicenow.com/index.php?oldid=240893> *Contributors:* Fuji.publishing.user, Joseph.messerschmidt

Attachment Creator Web Service *Source:* <http://wiki.servicenow.com/index.php?oldid=250040> *Contributors:* David Loo, John.andersen, John.ramos, Joseph.messerschmidt, Mark.stanger, Martin.wood, Publishing.user, Rachel.sienko, Steven.wood, Vaughn.romero

JSON *Source:* <http://wiki.servicenow.com/index.php?oldid=241023> *Contributors:* Emily.partridge, Fuji.publishing.user, John.ramos

CSV *Source:* <http://wiki.servicenow.com/index.php?oldid=250223> *Contributors:* David Loo, Emily.partridge, Fuji.publishing.user, John.ramos, Joseph.messerschmidt, Phillip.salzman, Rachel.sienko, Steven.wood, Vaughn.romero

EXCEL *Source:* <http://wiki.servicenow.com/index.php?oldid=250571> *Contributors:* David Loo, Emily.partridge, Fuji.publishing.user, John.ramos, Joseph.messerschmidt, Phillip.salzman, Rachel.sienko, Steven.wood

XML *Source:* <http://wiki.servicenow.com/index.php?oldid=251070> *Contributors:* David Loo, G.yedwab, John.ramos, Joseph.messerschmidt, Phillip.salzman, Rachel.sienko, Steven.wood, Vaughn.romero

PDF *Source:* <http://wiki.servicenow.com/index.php?oldid=250773> *Contributors:* David Loo, Emily.partridge, Fuji.publishing.user, John.ramos, Joseph.messerschmidt, Phillip.salzman, Publishing.user, Rachel.sienko, Steven.wood

RSS Feed Generator *Source:* <http://wiki.servicenow.com/index.php?oldid=250827> *Contributors:* CapaJC, David Loo, Don.Goodliffe, Guy.yedwab, Ishrath.razvi, Jared.laethem, Joe.Westrich, John.ramos, Joseph.messerschmidt, Peter.smith, Rachel.sienko, Rcade, Vhearne

RSS Feed Reader *Source:* <http://wiki.servicenow.com/index.php?oldid=248439> *Contributors:* Amy.bowman, David Loo, G.yedwab, Guy.yedwab, Joseph.messerschmidt, Mark.stanger, Rachel.sienko, Steven.wood

ODBC Driver *Source:* <http://wiki.servicenow.com/index.php?oldid=250761> *Contributors:* Cheryl.dolan, David Loo, David.Bailey, Emily.partridge, Fuji.publishing.user, Gadi.yedwab, Guy.yedwab, Ishrath.razvi, Joe.Westrich, Joe.zucker, John.andersen, John.ramos, Joseph.messerschmidt, Juell.solaegui, Mike.currie, Norris.merritt, Peter.smith, Phillip.salzman, Rachel.sienko, Steven.wood, Vaughn.romero

HTTP Client Connection Management *Source:* <http://wiki.servicenow.com/index.php?oldid=129169> *Contributors:* Emily.partridge, Joseph.messerschmidt, Steven.wood

Running Interactive SQL (ODBC) *Source:* <http://wiki.servicenow.com/index.php?oldid=230277> *Contributors:* Emily.partridge, John.ramos, Joseph.messerschmidt, Mark.stanger, Publishing.user, Steven.wood, Vaughn.romero

Using the ODBC Driver in Excel 2010 *Source:* <http://wiki.servicenow.com/index.php?oldid=245906> *Contributors:* Emily.partridge, John.ramos, Joseph.messerschmidt, Steven.wood, Suzanne.smith

Using the ODBC Driver in Crystal Reports 2008 *Source:* <http://wiki.servicenow.com/index.php?oldid=245908> *Contributors:* Emily.partridge, John.ramos, Joseph.messerschmidt, Mark.stanger, Steven.wood

Using ODBC Driver in SQL Server 2008 *Source:* <http://wiki.servicenow.com/index.php?oldid=245911> *Contributors:* Emily.partridge, John.ramos, Joseph.messerschmidt, Peter.smith, Steven.wood, Vaughn.romero

Image Sources, Licenses and Contributors

Image:Warning.gif *Source:* <http://wiki.servicenow.com/index.php?title=File:Warning.gif> *License:* unknown *Contributors:* CapaJC

Image:Web services app menu.png *Source:* http://wiki.servicenow.com/index.php?title=File:Web_services_app_menu.png *License:* unknown *Contributors:* Fuji.publishing.user

Image:choice_values.png *Source:* http://wiki.servicenow.com/index.php?title=File:Choice_values.png *License:* unknown *Contributors:* Vaughn.romero

Image:ElementFormDefaultProperty.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ElementFormDefaultProperty.png> *License:* unknown *Contributors:* Christen.mitchell

Image:Plugin.gif *Source:* <http://wiki.servicenow.com/index.php?title=File:Plugin.gif> *License:* unknown *Contributors:* CapaJC, Joseph.messerschmidt

Image:Ws iset.png *Source:* http://wiki.servicenow.com/index.php?title=File:Ws_iset.png *License:* unknown *Contributors:* David Loo

Image:System web services.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:System_web_services.jpg *License:* unknown *Contributors:* David Loo

Image:Ws iset iset.png *Source:* http://wiki.servicenow.com/index.php?title=File:Ws_iset_iset.png *License:* unknown *Contributors:* David Loo

Image:Edit web service.png *Source:* http://wiki.servicenow.com/index.php?title=File:Edit_web_service.png *License:* unknown *Contributors:* David Loo

Image:Create mapped web service.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Create_mapped_web_service.jpg *License:* unknown *Contributors:* David Loo

Image:Soap transform map.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_transform_map.jpg *License:* unknown *Contributors:* David Loo

Image:Soap notification.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_notification.jpg *License:* unknown *Contributors:* David Loo

File:soap_message_demo1.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_demo1.png *License:* unknown *Contributors:* David Loo

File:soap_message_function.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_function.png *License:* unknown *Contributors:* David Loo

File:soap_message_parameters.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_parameters.png *License:* unknown *Contributors:* David Loo

File:soap_message_test.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_test.png *License:* unknown *Contributors:* David Loo

File:soap_message_sample_script.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_sample_script.png *License:* unknown *Contributors:* David Loo, Fuji.publishing.user

File:soap_message_mid.png *Source:* http://wiki.servicenow.com/index.php?title=File:Soap_message_mid.png *License:* unknown *Contributors:* David Loo

File:scripted_web_service_module.png *Source:* http://wiki.servicenow.com/index.php?title=File:Scripted_web_service_module.png *License:* unknown *Contributors:* David Loo

Image:Sys web service.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Sys_web_service.jpg *License:* unknown *Contributors:* Ajandersen

Image:Input output.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Input_output.jpg *License:* unknown *Contributors:* Ajandersen

Image:GetProperty.jpg *Source:* <http://wiki.servicenow.com/index.php?title=File:GetProperty.jpg> *License:* unknown *Contributors:* Ajandersen

Image:InputOutputBlackberry.jpg *Source:* <http://wiki.servicenow.com/index.php?title=File:InputOutputBlackberry.jpg> *License:* unknown *Contributors:* Ajandersen

File:hierarchical_incident.png *Source:* http://wiki.servicenow.com/index.php?title=File:Hierarchical_incident.png *License:* unknown *Contributors:* David Loo

File:hierarchical_custom_comments.png *Source:* http://wiki.servicenow.com/index.php?title=File:Hierarchical_custom_comments.png *License:* unknown *Contributors:* David Loo

File:hierarchical_wsd11.png *Source:* http://wiki.servicenow.com/index.php?title=File:Hierarchical_wsd11.png *License:* unknown *Contributors:* David Loo

File:hierarchical_wsd12.png *Source:* http://wiki.servicenow.com/index.php?title=File:Hierarchical_wsd12.png *License:* unknown *Contributors:* David Loo

Image:GetBreadURL.png *Source:* <http://wiki.servicenow.com/index.php?title=File:GetBreadURL.png> *License:* unknown *Contributors:* Jared.laethem

Image:Rss format.png *Source:* http://wiki.servicenow.com/index.php?title=File:Rss_format.png *License:* unknown *Contributors:* David Loo

Image:Rss out.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Rss_out.jpg *License:* unknown *Contributors:* David Loo

Image:odbc_isql1.png *Source:* http://wiki.servicenow.com/index.php?title=File:Odbc_isql1.png *License:* unknown *Contributors:* David Loo, Joseph.messerschmidt

Image:odbc_isql2.png *Source:* http://wiki.servicenow.com/index.php?title=File:Odbc_isql2.png *License:* unknown *Contributors:* David Loo, Joseph.messerschmidt

File:Odbc_isql_shortcut_maxrows.png *Source:* http://wiki.servicenow.com/index.php?title=File:Odbc_isql_shortcut_maxrows.png *License:* unknown *Contributors:* Joseph.messerschmidt

File:ExcelOtherQuery.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ExcelOtherQuery.png> *License:* unknown *Contributors:* Juell.solaegui, Steven.wood

File:ExcelServiceNowDataSource.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ExcelServiceNowDataSource.png> *License:* unknown *Contributors:* Juell.solaegui, Steven.wood

File:ExcelDataSourceLogin.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ExcelDataSourceLogin.png> *License:* unknown *Contributors:* Juell.solaegui, Steven.wood

File:ExcelChooseODBCColumns.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ExcelChooseODBCColumns.png> *License:* unknown *Contributors:* Juell.solaegui, Steven.wood

Image:Excel_Query_Builder2.png *Source:* http://wiki.servicenow.com/index.php?title=File:Excel_Query_Builder2.png *License:* unknown *Contributors:* David Loo, Steven.wood

File:ExcelODBCResults.png *Source:* <http://wiki.servicenow.com/index.php?title=File:ExcelODBCResults.png> *License:* unknown *Contributors:* David Loo, Juell.solaegui, Steven.wood

File:crystal_standard.png *Source:* http://wiki.servicenow.com/index.php?title=File:Crystal_standard.png *License:* unknown *Contributors:* David Loo

File:crystal_create_connection.png *Source:* http://wiki.servicenow.com/index.php?title=File:Crystal_create_connection.png *License:* unknown *Contributors:* David Loo

File:crystal_select_table.png *Source:* http://wiki.servicenow.com/index.php?title=File:Crystal_select_table.png *License:* unknown *Contributors:* David Loo

File:crystal_select_fields.png *Source:* http://wiki.servicenow.com/index.php?title=File:Crystal_select_fields.png *License:* unknown *Contributors:* David Loo

File:crystal_report.png *Source:* http://wiki.servicenow.com/index.php?title=File:Crystal_report.png *License:* unknown *Contributors:* David Loo

Image:new_linked_server.png *Source:* http://wiki.servicenow.com/index.php?title=File:New_linked_server.png *License:* unknown *Contributors:* David Loo

Image:odbc_sqlserver_security.png *Source:* http://wiki.servicenow.com/index.php?title=File:Odbc_sqlserver_security.png *License:* unknown *Contributors:* David Loo

Image:sqlserver_provider_options.jpg *Source:* http://wiki.servicenow.com/index.php?title=File:Sqlserver_provider_options.jpg *License:* unknown *Contributors:* David Loo

Image:odbc_sqlserver_query.png *Source:* http://wiki.servicenow.com/index.php?title=File:Odbc_sqlserver_query.png *License:* unknown *Contributors:* David Loo