User guide for avnsSearch

Date created: 2016-09-19

Updated: 2016-09-19

Version: 1.1

Document information

Appendix

Appendix	Name	Path
1	RESTletGuide.pdf	
2	Search.pdf	
3		
4		
5		
6		
7		
8		

Revision

Date	Name	Version	Comments
2012-10-11		1.0	Document created
2016-09-07		1.1	Revised

Content

Introduction

Summary

This document describes how to use the AVNS Search functionality. AVNS Search is a method to be used by external systems to get information from Netsuite by calling a REST-function. Head components in the system are Saved Searches and a script that describes the implementation of the REST-function. A Saved Search defines what record type to get from the system and may also contain one or more filters to reduce the number of results. The Saved Search also describes what fields to be read on the record

The REST-call can contain additional filters to further reduce the number of results.

The REST function

The REST function is a flexible API used to communicate with Netsuite. Functions can be tailored to fit the customer needs. The Http call consists of two parts, header and body. The format for the body is based on JSON, this must be defined in the header by setting “Content-Type” to “application/json”.

In order to test a REST function one can use an application called Fiddler. Fiddler is a Windows application, but can be run in OS X using Wine. If that’s not an option one can install an application called “Advanced REST Client Application” (ARC) into the web browser Crome. ARC is though very sluggish, especially when replies contains many records.

Header (non token based autentification)

Calling the REST function is done by a http POST call were the autentification is done inside the http-header.

The correct web address for avnsSearch that should be called, can be found in Netsuite by going to Customization/Scripting/Script Deployment and look for avnsSearch. Open the deployment and look for EXTERNAL URL.

Header information that must be present is:

Host: rest.netsuite.com

Authorization: NLAuth nlauth_account=aaa, nlauth_email= zzz,nlauth_signature= xxx, nlauth_role=yyy

Content-Type: application/json; charset=UTF-8

Content-Length: nnn

Be careful when writing Authorization, it must be written exactly as above (except for values like aaa, zzz, yyy and xxx which should be replaced by correct values, see Figure 1). Content-Length is the number of characters in body.

Body

Example 1: The tiniest form of Body with no extra filters might look like this.

```
{
        "recordType": "transaction",
        "searchId": "customsearch_my_first_search",
        "filters": []
}
```

In the example 1 the saved search “customsearch_my_first_search” is a basic saved search on transactions with no filters attached. In the body of the call, “recordType” is set to transaction. As this is the same record type, the call will return all types of transactions. If we wanted only invoices we just have to set set “recordType” in body to “invoice” without having to change the saved search. Since the body also contains empty filters we will get a maximum of 10000 records. How to get any number of records will be explained later on in this document.

The call from Fiddler looks like this:

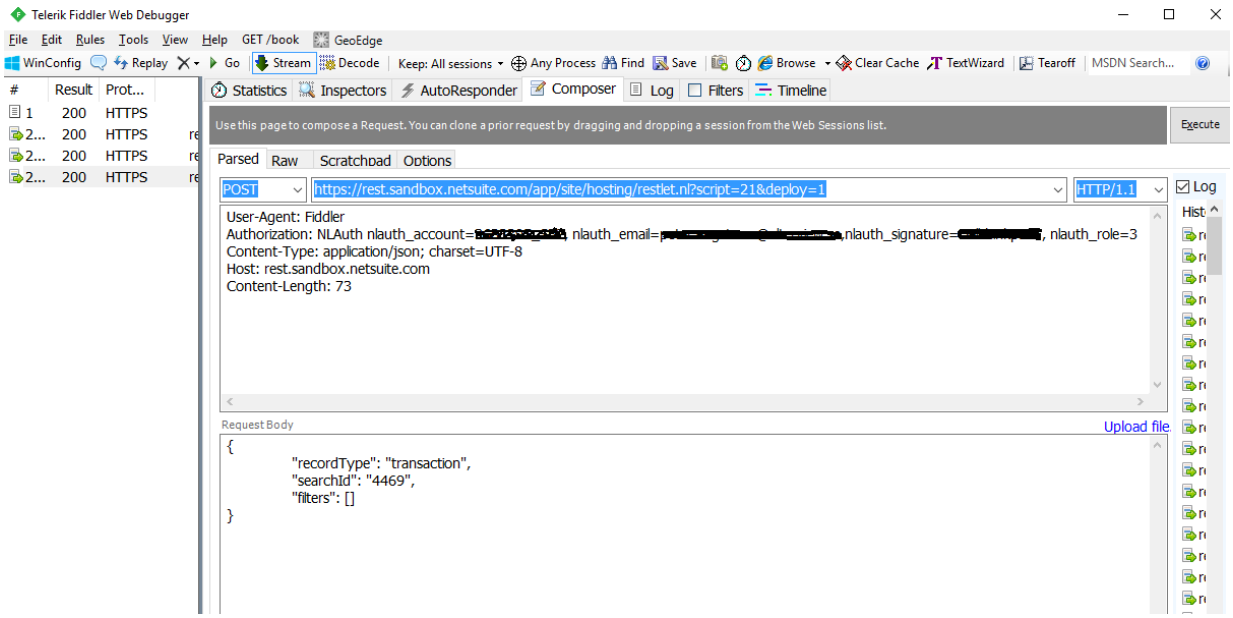
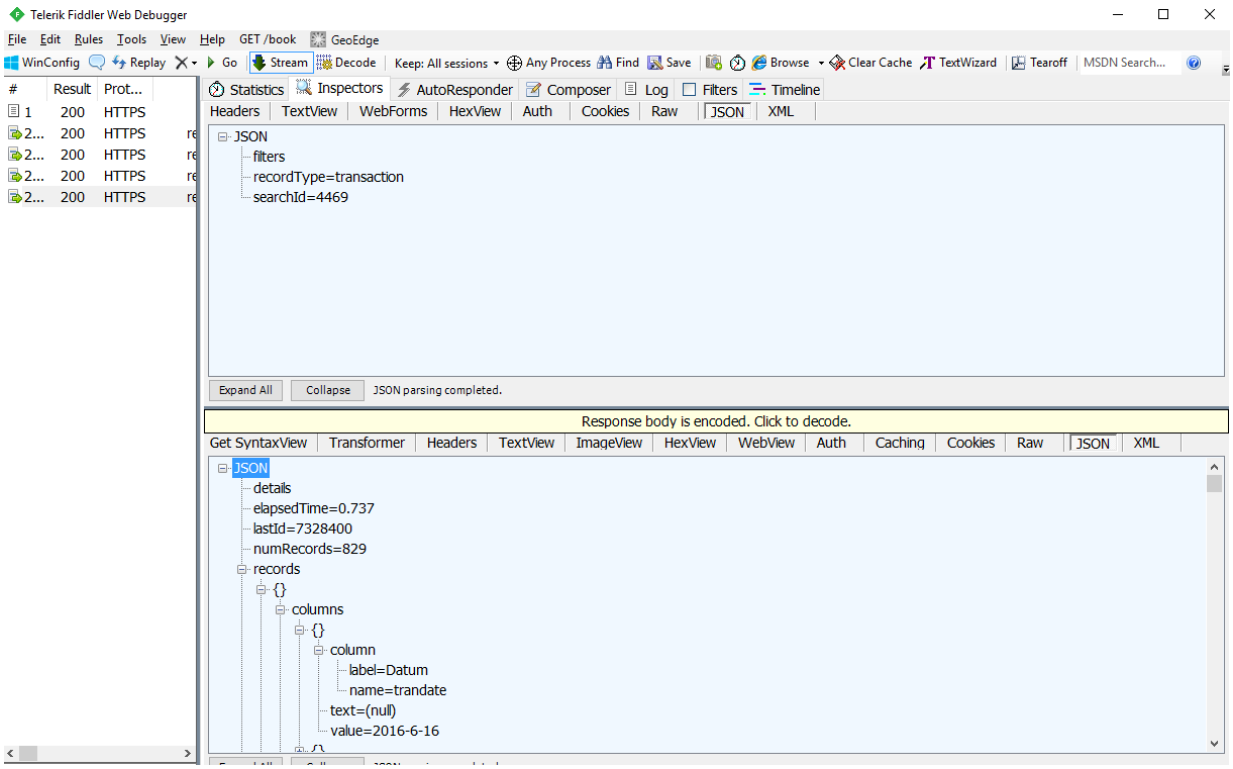


Figure 1: Fiddler call

The answer from avnsSearch from above call:



Exemple 2: A body with one filter:

```
{
        "recordType": "transaction",
        "searchId": "customsearch_my_first_search",
        "filters": [
                [{
                        "name": "trandate",
                        "operator": "after",
                        "value1": "2016-01-01"
                }]
        ]
}
```

As seen in the above examples the search id can be either the numerical value or the text value.

Filters are not mandatory in either the saved search or the body. The filters given in the body are added to the saved search e g filters defined in body will not replace those given in the saved search.

Mark that for filters containing dates, the format of the date must be in same format as the date format selected in settings by the calling user. User is the one that owns the e-mail address set in the header.

Exemple 3: A body with two filters that will filter out records where debet amount is greater than 2090 and the transaction date is after a certain date.

```
{
        "recordType": "transaction",
        "searchId": "customsearch_my_first_search",
        "maxRecords": "20",
        "filters": [
                [{
                        "name": "trandate",
                        "operator": "after",
                        "value1": "2016-01-01"
                }], "AND", [{
                        "name": "debitamount",
                        "operator": "greaterthan",
                        "value1": "2090"
                }]
        ]
}
```

Getting a large number of records

In order to get a large number of records or records containing a lot of data there are parameters to control the size of the returned result. The parameter “maxRecords” limit the amount of returned records, but there can be more records than specified by the parameter, so one can miss some results. If so, it’s better to use the parameters “page” and “pageSize”.

The parameter “pageSize” defines how many records that should be returned on each page. Default value for “pageSize” is 1000. The parameter “page” determine which page to get (mark that page numbering starts from zero).

Exemple 4: Get the result page by page.

```
{
        "recordType": "transaction",
        "searchId": "customsearch_my_first_search",
        "page": 0,
        "pageSize": 100,
        "filters": [
                [{
                        "name": "trandate",
                        "operator": "between",
                        "value1": "2010-01-01",
                        "value2": "2016-09-09"
                }]
        ]
}
```

In order to get all records using ”pageing” one must increment the ”page” value by one for each call until ”numRecords” is zero.

Exemple 5: An example using formula. Get all transactions where the tax amount is non integer.

```
{
  "recordType": "transaction",
  "searchId": "customsearch_my_first_search",
  "page": 0,
  "pageSize": 20,
  "filters": [
   [{
      "name": "formulanumeric:{taxamount}-TRUNC({taxamount},0)",
      "operator": "greaterthan",
      "value1": "0"
    }]
  ]
}
```

Parameters for body

Parameter	Comment
recordType	Record type. Allowed values can be found in Netsuite Record Browser.
Searchid	The saved search internal id. In the examples in the guide ”customsearch_my_first_search” or “4469”
maxRecords	Allowed values > 1. Default is 10000. This parameter is not used when using the parameter: ”page”
pageSize	Used to define page size. Allowable range is 1 - 4000. Default is 1000.
page	Tell avnsSearch what page to return. First page is number zero. To get all results increase page by one for each call until numRecords = 0. avnsSearch don’t look at the parameter “maxRecords” when “page” is used.
skiplabel	Don’t show labels in result if set to “T”.
skiptext	Don’t show text values if set to ”T”
skipname	Don’t show field name if set to ”T”
Filters	Filters to add to the saved search. This parameter is used add more filters to the original saved search. The syntax for creating filter objects follow the Netsuite documentation on “setFilterExpression” except that each filter in the array isn’t an array itself but a javascript object with parameters called: “name”, “operator” and “value1” / ”value2”

Operators for filters

In order to know what operators that can be used, search for ”Search Operators” in Netsuite Help Center. There are also some predefined date relative parameters that can be used for “value1” and “value2” when filtering on dates. These can be found by searching for “Search Date Filters” in Netsuite Help Center.

Result

Parameter	Comment
elapsedTime	Total processing time on server in seconds.
numRecords	Number of returned records.
lastId	The internalid on the last returned record for done search.
status	OK or ERROR. If ERROR the array called “details” contains one or more error messages.
details	An array containing error messages. Hopefully empty.
records	An array of records. Every record consists of 3 parameters (bold font): **Id**: Internalid for the record. **recordType**: Type of record. Types are described in Netsuite Help Center under “SuiteScript Supported Records”. **columns:** An array of search columns. Each column has the following parameters (italic font): *value*: Value. *text*: Textual value. (for fields of type List/Record). This can be turned off by setting the parameter “skiptext” to “T”. *column*: An object with two parameters(italic underlined): *name*: Name of the column. This can be turned off by setting the parameter “skipname” to “T”. *label*: Column heading. This can be turned off by setting the parameter “skipname” to “T”.