m_utl_json2ds.sas File Reference

Utilities

Utility macro to export a JSON file into SAS datasets or tables

---

Description

This program converts a JSON javascript object notation file into SAS datasets or database tables. The macro will process the json file looking for key value pairs and arrays. A SAS dataset or database table is created containing all key value pairs and is named by the _OUT_DS_ value. Furthermore complementary tables are created for each array found in the json file. These arrays are named by the following name convention: _OUT_DS_ARRAY-NAME_.
In case of a json file that contains no pairs and only an array, only one table will be created with the given _OUT_DS_ name. The program handles common differences of the _PRETTY_ format by removing existing formats, and reformatting again by default. The following JSON data types are implemented:
Number: a signed decimal number.
String: a sequence of zero or more Unicode characters.
Boolean: either of the values true or false.
Array: an ordered list of zero or more values.
Object: an unordered collection of key-value pairs
Null: an empty value, using the word null.

Note

Arrays are allowed up to two levels deep, with no restriction on the number of arrays. Both _key-value_ and _ordered-list_ type arrays are allowed. A Boolean is handled as numeric 0 or 1 value with a character format with the values _true_ or _false_, and is set to _null_ if the value is missing. A _Null_ value is currently only supported as a missing value representation for a character type value. Key values longer than the 32 positions restriction in SAS are truncated to 32 positions with a suffix. The full key value is kept in the attribute label. For details see the result of Example-2 "age_is_too_long_for_sas_to_handle".

Autors

Paul Alexander Canals y Trocha (paul.canals@gmail.com)

Date

2022-11-08 00:00:00

Version

22.1.11

Link

https://github.com/paul-canals/toolbox

Parameters

Input	help	Parameter, if set (Help or ?) to print the Help information in the log. In all other cases this parameter should be left out from the macro call.
Input	json_file	Specifies the name and location of the javascript object notation (JSON) file. The default value is: _NONE_.
Input	out_ds	Specifies the full LIBNAME.TABLENAME name of the output SAS dataset or database table. The default value is: _NONE_.
Input	pretty	Boolean [Y|N] parameter to specify to reformat the JSON file structure. The argument PRETTY=N will create a more human-readable format using indention to illustrate the JSON structure. The argument PRETTY=Y will expect that the json file is already formatted correctly. The default value for PRETTY is N (always reformat).
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- Output SAS dataset or database table.

Calls

- m_utl_nlobs.sas
- m_utl_print_message.sas
- m_utl_print_mtrace.sas

Usage

Example 1: Show help information:

```
%m_utl_json2ds(?)
```

Example 2: Step 1 - Create a JSON file representing a person with no arrays:

```
filename person "%sysfunc(getoption(WORK))/person.json";

data _null_;
   file person;
   put '{';
   put '  "firstName": "John",';
   put '  "lastName": "Smith",';
   put '  "isAlive": true,';
   put '  "age_is_too_long_for_sas_to_handle": 25,';
   put '  "address": {';
   put '    "streetAddress": "21 2nd Street",';
   put '    "city": "New York",';
   put '    "state": "NY",';
   put '    "postalCode": "10021-3100"';
   put '  },';
   put '  "phoneNumber": "212 555-1234"';
   put '  "children": false,';
   put '  "spouse": null';
   put '}';
run;

data _null_;
   infile person;
   input;
   put _infile_;
run;
```

Example 2: Step 2 - Convert person.json file into WORK.person:

```
%m_utl_json2ds(
   json_file = %quote(%sysfunc(getoption(WORK))/person.json)
 , out_ds    = person
 , pretty    = Y
 , debug     = Y
   );

title 'Person';
proc print data=WORK.person;
run;
```

Example 3: Step 1 - Create a JSON file representing a person with arrays:

```
filename person "%sysfunc(getoption(WORK))/person.json";

data _null_;
   file person;
   put '{';
   put '  "firstName": "John",';
   put '  "lastName": "Smith",';
   put '  "isAlive": true,';
   put '  "age": 25,';
   put '  "address": {';
   put '    "streetAddress": "21 2nd Street",';
   put '    "city": "New York",';
   put '    "state": "NY",';
   put '    "postalCode": "10021-3100"';
   put '  },';
   put '  "phoneNumbers": [';
   put '    [';
   put '      "type",';
   put '      "number"';
   put '    ],';
   put '    [';
   put '      "home",';
   put '      "212 555-1234"';
   put '    ],';
   put '    [';
   put '      "office",';
   put '      "646 555-4567"';
   put '    ],';
   put '    [';
   put '      "mobile",';
   put '      "123 456-7890"';
   put '    ]';
   put '  ],';
   put '  "children": [';
   put '    {';
   put '      "type": "daughter",';
   put '      "name": "Susanne"';
   put '    },';
   put '    {';
   put '      "type": "son",';
   put '      "name": "William"';
   put '    },';
   put '  ],';
   put '  "spouse": null';
   put '}';
run;

data _null_;
   infile person;
   input;
   put _infile_;
run;
```

Example 3: Step 2 - Convert person.json file into WORK.person:

```
%m_utl_json2ds(
   json_file = %quote(%sysfunc(getoption(WORK))/person.json)
 , out_ds    = person
 , pretty    = Y
 , debug     = N
   );

title 'Person';
proc print data=WORK.person;
run;

title 'Person_Children';
proc print data=WORK.person_children;
run;

title 'Person_Phonenumbers';
proc print data=WORK.person_phonenumbers;
run;
```

Example 4: Step 1 - Create a JSON file representing a person using wrong json format:

```
filename person "%sysfunc(getoption(WORK))/person.json";

data _null_;
   file person;
   put '{';
   put '  "firstName": "John"';
   put ' ,"lastName": "Smith"';
   put ' ,"isAlive": true';
   put ' ,"age": 25';
   put ' ,"address": {';
   put '    "streetAddress": "21 2nd Street"';
   put '   ,"city": "New York"';
   put '   ,"state": "NY"';
   put '   ,"postalCode": "10021-3100"';
   put '  }';
   put ' ,"phoneNumbers": [';
   put ' ]';
   put ' ,"children": [';
   put ' ]';
   put ' ,"spouse": null';
   put '}';
run;

data _null_;
   infile person;
   input;
   put _infile_;
run;
```

Example 4: Step 2 - Reformat, and convert person.json file into WORK.person:

```
%m_utl_json2ds(
   json_file = %quote(%sysfunc(getoption(WORK))/person.json)
 , out_ds    = person
 , pretty    = N
 , debug     = Y
   );

title 'Person';
proc print data=WORK.person;
run;
```

Copyright