Article
Sean Connelly · May 31, 2017 28m read

Cogs Library

Cogs Library

Over the next few months I will be releasing a number of open source libraries and tools to the Caché community.

Most of the code has evolved from previous production grade solutions over the years and I am collating it together under a single overarching library package that I am calling Cogs.

I will be releasing it in stages as I complete the documentation for each logical release.

Before each full release I will be making each library / tool available on pre-release evaluation and I am open to anyone wanting to participate and provide early feedback.

The core Cogs library will include...

- A comprehensive set of JSON functions
- A JSON-RPC gateway with RAD development features
- Code generation of JavaScript proxy objects
- Code generation of TypeScript proxy objects with object types and auto completion
- CoffeeTable, an ORM JavaScript client library that is compatible with Caché, PostgreSQL and MongoDB
- Fast and scalable Node.JS connector for Caché (currently experimental)

As well as the above I have been working on a number of tools to make open source development easier and plan to release these under the same Cogs library. These tools include...

- An easy to use unit test library with a web interface
- A markdown documentation editor with live preview that saves documents directly into XDATA source code. These documents live inside Caché and can be previewed from Studio with a single click. The markdown compiler includes mermaid for quick production of diagrams (great for Ensemble developers wanting to document message flows).
- A simple code sync tool for working with Git

Kicking off the first release I have decided to pre-release the Cogs JSON library within the next 10 days. Please get in contact if you are interested in this release.

As a flavour of what to expect, the JSON library has the following features...

- Serialise JSON from Cache objects, with features such as JSON name decorators and calculated JSON value functions
- De-serialisation from JSON to Cache objects, including directly to persistent classes.
- Serialise JSON to arrays and globals
- De-serialise JSON from arrays and globals
- Streamlined SQL resultset serialiser that outputs directly to device
- Special data types for storage of raw JSON
- Handle any combination of type, object, array and list in any combination of nested types
- Special JSON mixer variant to serialise and de-serialise objects of mixed object types.

Example

Extend any class with Cogs.JsonClass...

```
Class Example.Person Extends (%Persistent, Cogs.JsonClass)
{

Property FirstName As %String;

Property LastName As %String;

/// @JSONNAME=BirthDate
Property DateOfBirth As %Date;

/// @JSONIGNORE
Property Secret As %String;

Property Hobbies As list Of %String;

}
```

take some JSON...

```
{
   "BirthDate":"1970-03-25",
   "FirstName":"Sean",
   "Hobbies":[
      "Photography",
      "Walking",
      "Football"
   ],
   "LastName":"Connelly"
}
```

parse and save it...

```
set person=##class(Example.Person).parseJSON(json)
set sc=person.%Save()
```

notice that the library automatically converts date types (as well as boolean, number, null and undefined)

```
person=<OBJECT REFERENCE>[1@Example.Person]
+----------------- general information ---------------
|      oref value: 1
|      class name: Example.Person
| reference count: 2
+----------------- attribute values ------------------
|       %Concurrency = 1  <Set>
|        DateOfBirth = 47200
|          FirstName = "Sean"
|           LastName = "Connelly"
|             Secret = ""
+----------------- swizzled references ---------------
|          i%Hobbies = ""
|       i%Hobbies(1) = "Photography"
```

```
|       i%Hobbies(2) = "Walking"
|       i%Hobbies(3) = "Football"
|          r%Hobbies = "2@%Collection.ListOfDT"  <Set>
+-----------------------------------------------------
```

now open the persistent object and call its toJSON() method...

```
set person=##class(Example.Person).%OpenId(1)
write !,person.toJSON()
```

and its as easy as that...

```
{"BirthDate":"1970-03-25","FirstName":"Sean","Hobbies":["Photography","Walking","Foot
ball"],"LastName":"Connelly"}
```

How fast is it?

The above parseJSON() example will complete in 1/20th of a millisecond whilst its toJSON() will complete in 1/50th of a millisecond

How stable is it?

The core JSON library has evolved from numerous developments and re-writes over the years. Some of these versions underpin numerous hospital applications. The Cogs release is the latest evolution, highly optimised and fully unit tested.

This includes throwing some pretty gnarly JSON at it...

```
{
   "TestAllAsciiChars":" !\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[
\\]Ž••‘’“”•–—˜™š•žŸ ¡¢£¤¥¦§¨©ª«¬-
®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ",
   "TestArrayOfBoolean":{
      "ALPHA":true,
      "CHARLIE":false,
      "DELTA":null,
      "ECHO":false,
      "FOXTROT":true
   },
   "TestArrayOfDate":{
      "Happy-Horolog :)":"1840-12-31",
      "doB":"2126-12-06",
      "nully":null
   },
   "TestArrayOfInteger":{
      "A THOUSAND!":1000,
      "NINETY NINE":99,
      "ONE":1,
      "THREE":3,
      "TWO":2,
      "WORLD DEBT $":59546526326827,
      "WORLD WORTH $":241000000000000,
      "nully":0
```

```
},
"TestArrayOfObject":{

},
"TestArrayOfString":{
   "City":"London W1",
   "Country":"UK",
   "Detective":"Sherlock Holmes",
   "District":"Marylebone",
   "FOOBAR":"{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
   "Street":"221B Baker Street",
   "b4d,Th1ng5!":"\\\"\\\"/\"\b\t\r\f\n",
   "not-nully":"",
   "nully":null
},
"TestBooleanNo":false,
"TestBooleanYes":true,
"TestDate":"2017-05-31",
"TestEmptyArrayOfBoolean":{

},
"TestEmptyArrayOfDate":{

},
"TestEmptyArrayOfInteger":{

},
"TestEmptyArrayOfString":{

},
"TestEmptyListOfBoolean":[

],
"TestEmptyListOfOfDate":[

],
"TestEmptyListOfOfInteger":[

],
"TestEmptyListOfString":[

],
"TestEscapes":"\\\"\\\"/\"\b\t\r\f\n",
"TestInteger":42,
"TestListOfBoolean":[
   true,
   false,
   false,
   null,
   true,
   false,
   true,
   true,
   false
],
"TestListOfObject":[

],
"TestListOfOfDate":[
```

```
      "1840-12-31",
      "1841-01-10",
      "1841-04-10",
      "1843-09-27",
      null,
      "1868-05-18",
      "2114-10-16",
      "4578-11-27"
   ],
   "TestListOfOfInteger":[
      0,
      43564356546456,
      0.345,
      0.2,
      0,
      -464356.75675,
      76567,
      3.141592653589793238,
      7653475667,
      -65753,
      45676.56,
      -6807.383887076754324
   ],
   "TestListOfString":[
      "Sherlock Holmes",
      "221B Baker Street",
      "Marylebone",
      "London W1",
      "UK",
      null,
      "{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
      "\\\"\\\"/\"\b\t\r\f\n",
      "0",
      "1",
      "345345345345345345400"
   ],
   "TestLongNumber":499479768050558758400000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000,
   "TestLongString":"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Du
is aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat
nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui off
icia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis iste natus e
rror sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa
quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicab
o. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed qu
ia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro
quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed
 quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaer
at voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis s
uscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iur
e reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel
illum qui dolorem eum fugiat quo voluptas nulla pariatur?",
   "TestNotBoolean1":"true",
   "TestNotBoolean2":"false",
   "TestNullBoolean":null,
   "TestNullDate":null,
   "TestNullInteger":0,
```

```
   "TestNullString":null,
   "TestNullTimestamp":null,
   "TestRawJson":{
      "TestAllAsciiChars":" !\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWX
YZ[\\]Ž••‘’“”•–—˜™š•žŸ ¡¢£¤¥¦§¨©ª«¬-
®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ",
      "TestArrayOfBoolean":{
         "ALPHA":true,
         "CHARLIE":false,
         "DELTA":null,
         "ECHO":false,
         "FOXTROT":true
      },
      "TestArrayOfDate":{
         "Happy-Horolog :)":"1840-12-31",
         "doB":"2126-12-06",
         "nully":null
      },
      "TestArrayOfInteger":{
         "A THOUSAND!":1000,
         "NINETY NINE":99,
         "ONE":1,
         "THREE":3,
         "TWO":2,
         "WORLD DEBT $":59546526326827,
         "WORLD WORTH $":241000000000000,
         "nully":0
      },
      "TestArrayOfObject":{

      },
      "TestArrayOfString":{
         "City":"London W1",
         "Country":"UK",
         "Detective":"Sherlock Holmes",
         "District":"Marylebone",
         "FOOBAR":"{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
         "Street":"221B Baker Street",
         "b4d,Th1ng5!":"\\\"\\\"/\"\b\t\r\f\n",
         "not-nully":"",
         "nully":null
      },
      "TestBooleanNo":false,
      "TestBooleanYes":true,
      "TestDate":"2017-05-31",
      "TestEmptyArrayOfBoolean":{

      },
      "TestEmptyArrayOfDate":{

      },
      "TestEmptyArrayOfInteger":{

      },
      "TestEmptyArrayOfString":{

      },
      "TestEmptyListOfBoolean":[
```

```
],
"TestEmptyListOfOfDate":[

],
"TestEmptyListOfOfInteger":[

],
"TestEmptyListOfString":[

],
"TestEscapes":"\\\"\\\"/\"\b\t\r\f\n",
"TestInteger":42,
"TestListOfBoolean":[
   true,
   false,
   false,
   null,
   true,
   false,
   true,
   true,
   false
],
"TestListOfObject":[

],
"TestListOfOfDate":[
   "1840-12-31",
   "1841-01-10",
   "1841-04-10",
   "1843-09-27",
   null,
   "1868-05-18",
   "2114-10-16",
   "4578-11-27"
],
"TestListOfOfInteger":[
   0,
   43564356546456,
   0.345,
   0.2,
   0,
   -464356.75675,
   76567,
   3.141592653589793238,
   7653475667,
   -65753,
   45676.56,
   -6807.383887076754324
],
"TestListOfString":[
   "Sherlock Holmes",
   "221B Baker Street",
   "Marylebone",
   "London W1",
   "UK",
   null,
   "{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
   "\\\"\\\"/\"\b\t\r\f\n",
```

```
         "0",
         "1",
         "345345345345345345400"
      ],
      "TestLongNumber":4994797680505587584000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000,
      "TestLongString":"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim venia
m, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
 Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugi
at nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis iste natu
s error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ip
sa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt expli
cabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed
 quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque por
ro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit,
sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam qu
aerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corpori
s suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum
iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, v
el illum qui dolorem eum fugiat quo voluptas nulla pariatur?",
      "TestNotBoolean1":"true",
      "TestNotBoolean2":"false",
      "TestNullBoolean":null,
      "TestNullDate":null,
      "TestNullInteger":0,
      "TestNullString":null,
      "TestNullTimestamp":null,
      "TestRawJson":{
         "data":{
            "TestAllAsciiChars":" !\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQR
STUVŽ••‘’“”•–—˜™š•žŸ ¡¢£¤¥¦§¨©ª«¬-
®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ",
            "TestArrayOfBoolean":{
               "ALPHA":true,
               "CHARLIE":false,
               "DELTA":null,
               "ECHO":false,
               "FOXTROT":true
            },
            "TestArrayOfDate":{
               "Happy-Horolog :)":"1840-12-31",
               "doB":"2125-11-27",
               "nully":null
            },
            "TestArrayOfInteger":{
               "A THOUSAND!":1000,
               "NINETY NINE":99,
               "ONE":1,
               "THREE":3,
               "TWO":2,
               "WORLD DEBT $":59546526326827,
               "WORLD WORTH $":241000000000000,
               "nully":null
            },
            "TestArrayOfObject":{

            },
```

```
"TestArrayOfString":{
   "City":"London W1",
   "Country":"UK",
   "Detective":"Sherlock Holmes",
   "District":"Marylebone",
   "FOOBAR":"{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
   "Street":"221B Baker Street",
   "b4d,Th1ng5!":"\\\"\\\"/\"\b\t\r\f\n",
   "not-nully":"",
   "nully":null
},
"TestBooleanNo":false,
"TestBooleanYes":true,
"TestDate":"2016-05-22",
"TestEmptyArrayOfBoolean":{

},
"TestEmptyArrayOfDate":{

},
"TestEmptyArrayOfInteger":{

},
"TestEmptyArrayOfString":{

},
"TestEmptyListOfBoolean":[

],
"TestEmptyListOfOfDate":[

],
"TestEmptyListOfOfInteger":[

],
"TestEmptyListOfString":[

],
"TestEscapes":"\\\"\\\"/\"\b\t\r\f\n",
"TestInteger":42,
"TestListOfBoolean":[
   true,
   false,
   false,
   null,
   true,
   false,
   true,
   true,
   false
],
"TestListOfObject":[

],
"TestListOfOfDate":[
   "1840-12-31",
   "1841-01-10",
   "1841-04-10",
   "1843-09-27",
```

```
               null,
               "1868-05-18",
               "2114-10-16",
               "4578-11-27"
            ],
            "TestListOfOfInteger":[
               "0",
               43564356546456,
               0.345,
               0.2,
               null,
               -464356.75675,
               76567,
               3.141592653589793238,
               7653475667,
               -65753,
               45676.56,
               -6807.383887076754324
            ],
            "TestListOfString":[
               "Sherlock Holmes",
               "221B Baker Street",
               "Marylebone",
               "London W1",
               "UK",
               null,
               "{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}",
               "\\\"\\\"/\"\b\t\r\f\n",
               "0",
               "1",
               "3453453453453453454 00"
            ],
            "TestLongNumber":4994797680505587584000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000,
            "TestLongString":"Lorem ipsum dolor sit amet, consectetur adipiscing elit
, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
 veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo cons
equat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore e
u fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culp
a qui officia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis ist
e natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, ea
que ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt
 explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugi
t, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neq
ue porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci v
elit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliq
uam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam c
orporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem ve
l eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequa
tur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?",
            "TestNotBoolean1":"true",
            "TestNotBoolean2":"false",
            "TestNullBoolean":null,
            "TestNullDate":null,
            "TestNullInteger":null,
            "TestNullString":null,
            "TestNullTimestamp":null,
            "TestRawJson":"",
```

```
            "TestSingleObject":"",
            "TestString":"{[{\\}[/][{[\"FOO\",\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\",\"BA
R\"]}]{\\}[/]]}",
            "TestStringNotNull":"",
            "TestStringOfJSON":"{\"menu\": { \"id\": \"file\", \"value\": \"File\", \
"popup\": { \"menuitem\": [ {\"value\": \"New\", \"onclick\": \"CreateNewdoc()\"}, {\
"value\": \"Open\", \"onclick\": \"Opendoc()\"}, {\"value\": \"Close\", \"onclick\":
\"Closedoc()\"} ] }}}",
            "TestTimestamp":"1966-01-27T23:12:02",
            "TestTimestampShort":"1966-01-27T00:00:00",
            "_id":"FOO"
         }
      },
      "TestSingleObject":"",
      "TestString":"{[{\\}[/][{[\"FOO\",\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\",\"BAR\"]}]
{\\}[/]]}",
      "TestStringNotNull":"",
      "TestStringOfJSON":"{\"menu\": { \"id\": \"file\", \"value\": \"File\", \"popup
\": { \"menuitem\": [ {\"value\": \"New\", \"onclick\": \"CreateNewdoc()\"}, {\"value
\": \"Open\", \"onclick\": \"Opendoc()\"}, {\"value\": \"Close\", \"onclick\": \"Clos
edoc()\"} ] }}}",
      "TestTime":"01:14:04",
      "TestTimestamp":"1966-01-27T23:12:02",
      "TestTimestampShort":"1966-01-27T00:00:00",
      "TestZero":0,
      "ZTestJsonMethod":"LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT, SED
 DO EIUSMOD TEMPOR INCIDIDUNT UT LABORE ET DOLORE MAGNA ALIQUA. UT ENIM AD MINIM VENI
AM, QUIS NOSTRUD EXERCITATION ULLAMCO LABORIS NISI UT ALIQUIP EX EA COMMODO CONSEQUAT
. DUIS AUTE IRURE DOLOR IN REPREHENDERIT IN VOLUPTATE VELIT ESSE CILLUM DOLORE EU FUG
IAT NULLA PARIATUR. EXCEPTEUR SINT OCCAECAT CUPIDATAT NON PROIDENT, SUNT IN CULPA QUI
 OFFICIA DESERUNT MOLLIT ANIM ID EST LABORUM. SED UT PERSPICIATIS UNDE OMNIS ISTE NAT
US ERROR SIT VOLUPTATEM ACCUSANTIUM DOLOREMQUE LAUDANTIUM, TOTAM REM APERIAM, EAQUE I
PSA QUAE AB ILLO INVENTORE VERITATIS ET QUASI ARCHITECTO BEATAE VITAE DICTA SUNT EXPL
ICABO. NEMO ENIM IPSAM VOLUPTATEM QUIA VOLUPTAS SIT ASPERNATUR AUT ODIT AUT FUGIT, SE
D QUIA CONSEQUUNTUR MAGNI DOLORES EOS QUI RATIONE VOLUPTATEM SEQUI NESCIUNT. NEQUE PO
RRO QUISQUAM EST, QUI DOLOREM IPSUM QUIA DOLOR SIT AMET, CONSECTETUR, ADIPISCI VELIT,
 SED QUIA NON NUMQUAM EIUS MODI TEMPORA INCIDUNT UT LABORE ET DOLORE MAGNAM ALIQUAM Q
UAERAT VOLUPTATEM. UT ENIM AD MINIMA VENIAM, QUIS NOSTRUM EXERCITATIONEM ULLAM CORPOR
IS SUSCIPIT LABORIOSAM, NISI UT ALIQUID EX EA COMMODI CONSEQUATUR? QUIS AUTEM VEL EUM
 IURE REPREHENDERIT QUI IN EA VOLUPTATE VELIT ESSE QUAM NIHIL MOLESTIAE CONSEQUATUR,
VEL ILLUM QUI DOLOREM EUM FUGIAT QUO VOLUPTAS NULLA PARIATUR?",
      "_id":"FOO"
   },
   "TestSingleObject":"",
   "TestString":"{[{\\}[/][{[\"FOO\",\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\"\",\"BAR\"]}]{\\
}[/]]}",
   "TestStringNotNull":"",
   "TestStringOfJSON":"{\"menu\": { \"id\": \"file\", \"value\": \"File\", \"popup\":
 { \"menuitem\": [ {\"value\": \"New\", \"onclick\": \"CreateNewdoc()\"}, {\"value\":
 \"Open\", \"onclick\": \"Opendoc()\"}, {\"value\": \"Close\", \"onclick\": \"Closedo
c()\"} ] }}}",
   "TestTime":"01:14:04",
   "TestTimestamp":"1966-01-27T23:12:02",
   "TestTimestampShort":"1966-01-27T00:00:00",
   "TestZero":0,
   "ZTestJsonMethod":"LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT, SED DO
 EIUSMOD TEMPOR INCIDIDUNT UT LABORE ET DOLORE MAGNA ALIQUA. UT ENIM AD MINIM VENIAM,
 QUIS NOSTRUD EXERCITATION ULLAMCO LABORIS NISI UT ALIQUIP EX EA COMMODO CONSEQUAT. D
UIS AUTE IRURE DOLOR IN REPREHENDERIT IN VOLUPTATE VELIT ESSE CILLUM DOLORE EU FUGIAT
```

```
 NULLA PARIATUR. EXCEPTEUR SINT OCCAECAT CUPIDATAT NON PROIDENT, SUNT IN CULPA QUI OF
FICIA DESERUNT MOLLIT ANIM ID EST LABORUM. SED UT PERSPICIATIS UNDE OMNIS ISTE NATUS
ERROR SIT VOLUPTATEM ACCUSANTIUM DOLOREMQUE LAUDANTIUM, TOTAM REM APERIAM, EAQUE IPSA
 QUAE AB ILLO INVENTORE VERITATIS ET QUASI ARCHITECTO BEATAE VITAE DICTA SUNT EXPLICA
BO. NEMO ENIM IPSAM VOLUPTATEM QUIA VOLUPTAS SIT ASPERNATUR AUT ODIT AUT FUGIT, SED Q
UIA CONSEQUUNTUR MAGNI DOLORES EOS QUI RATIONE VOLUPTATEM SEQUI NESCIUNT. NEQUE PORRO
 QUISQUAM EST, QUI DOLOREM IPSUM QUIA DOLOR SIT AMET, CONSECTETUR, ADIPISCI VELIT, SE
D QUIA NON NUMQUAM EIUS MODI TEMPORA INCIDUNT UT LABORE ET DOLORE MAGNAM ALIQUAM QUAE
RAT VOLUPTATEM. UT ENIM AD MINIMA VENIAM, QUIS NOSTRUM EXERCITATIONEM ULLAM CORPORIS
SUSCIPIT LABORIOSAM, NISI UT ALIQUID EX EA COMMODI CONSEQUATUR? QUIS AUTEM VEL EUM IU
RE REPREHENDERIT QUI IN EA VOLUPTATE VELIT ESSE QUAM NIHIL MOLESTIAE CONSEQUATUR, VEL
 ILLUM QUI DOLOREM EUM FUGIAT QUO VOLUPTAS NULLA PARIATUR?",
   "_id":"FOO"
}
```

To prove its all real, here is an object dump of the above JSON after it was parsed...

```
o=<OBJECT REFERENCE>[20@Cogs.Tests.Json.JsonClassTest]
+----------------- general information ---------------
|      oref value: 20
|      class name: Cogs.Tests.Json.JsonClassTest
| reference count: 2
+----------------- attribute values ------------------
|  TestAllAsciiChars = " !""#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[
\]^_`abcdefghijklmnopqrstuvwxyz{|}~"_$c(127,128,129,130,131,132,133,134,135,136,137,1
38,139,140,141,142,143,144,145,146,147,148,149,150,151,152,153,154,155,156,157,158,15
9)_" ¡¢£¤¥¦§¨©ª«¬-
®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ"
|      TestBooleanNo = 0
|     TestBooleanYes = 1
|           TestDate = 64434
|        TestEscapes = "\""\""/""""_$c(8,9,10,12,13)
|     TestIgnoreFlag = "WARNING, THIS IS PRIVATE AND SHOULD NOT BE SERIALISED INTO JS
ON!"
|        TestInteger = 42
|     TestLongNumber = 4994797680505587584000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
|     TestLongString = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim venia
m, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
 Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugi
at nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis iste natu
s error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ip
sa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt expli
cabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed
 quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque por
ro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit,
sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam qu
aerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corpori
s suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum
iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, v
el illum qui dolorem eum fugiat quo voluptas nulla pariatur?"
|    TestNotBoolean1 = "true"
|    TestNotBoolean2 = "false"
|    TestNullBoolean = ""
```

```
|       TestNullDate = ""
|    TestNullInteger = ""
|     TestNullString = $c(0)
|  TestNullTimestamp = ""
|        TestRawJson = "{""TestAllAsciiChars"":"" !\""#$%&'()*+,-./0123456789:;<=>?@A
BCDEFGHIJKLMNOPQRSTUVWXYZ[\\]"_$c(142,143,144,145,146,147,148,149,150,151,152,153,154
,157,158,159)_" ¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæ
çèéêëìíîïðñòóôõö÷øùúûüýþÿ"",""TestArrayOfBoolean"":{""ALPHA"":true,""CHARLIE"":false,
""DELTA"":null,""ECHO"":false,""FOXTROT"":true},""TestArrayOfDate"":{""Happy-Horolog
:)"":""1840-12-31"",""doB"":""2126-12-06"",""nully"":null},""TestArrayOfInteger"":{""
A THOUSAND!"":1000,""NINETY NINE"":99,""ONE"":1,""THREE"":3,""TWO"":2,""WORLD DEBT $"
":59546526326827,""WORLD WORTH $"":241000000000000,""nully"":0},""TestArrayOfObject""
:{},""TestArrayOfString"":{""City"":""London W1"",""Country"":""UK"",""Detective"":""
Sherlock Holmes"",""District"":""Marylebone"",""FOOBAR"":""{[{\\}[/][{[FOO,BAR]}]{\\}
[/]]}"",""Street"":""221B Baker Street"",""b4d,Th1ng5!"":""\\\""\\\""/\""\b\t\r\f\n""
,""not-nully"":""'"",""nully"":null},""TestBooleanNo"":false,""TestBooleanYes"":true,"
"TestDate"":""2017-05-31"",""TestEmptyArrayOfBoolean"":{},""TestEmptyArrayOfDate"":{}
,""TestEmptyArrayOfInteger"":{},""TestEmptyArrayOfString"":{},""TestEmptyListOfBoolea
n"":[],""TestEmptyListOfOfDate"":[],""TestEmptyListOfOfInteger"":[],""TestEmptyListOf
String"":[],""TestEscapes"":""\\\""\\\""/\""\b\t\r\f\n"",""TestInteger"":42,""TestLis
tOfBoolean"":[true,false,false,null,true,false,true,true,false],""TestListOfObject"":
[],""TestListOfOfDate"":[""1840-12-31"",""1841-01-10"",""1841-04-10"",""1843-09-27"",
null,""1868-05-18"",""2114-10-16"",""4578-11-27""],""TestListOfOfInteger"":[0,4356435
6546456,0.345,0.2,0,-464356.75675,76567,3.141592653589793238,7653475667,-65753,45676.
56,-6807.383887076754324],""TestListOfString"":[""Sherlock Holmes"",""221B Baker Stre
et"",""Marylebone"",""London W1"",""UK"",null,""{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}"",""\
\\""\\\""/\""\b\t\r\f\n"",""0"",""1"",""34534534534534534400""],""TestLongNumber"":4
9947976805055875840000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000,""TestLongString"":""Lor
em ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididun
t ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitatio
n ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in r
eprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteu
r sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit ani
m id est laborum. Sed ut perspiciatis unde omnis iste natus error sit voluptatem accu
santium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore v
eritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam volup
tatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni do
lores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolore
m ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius m
odi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim a
d minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi
 ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in e
a voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fu
giat quo voluptas nulla pariatur?"",""TestNotBoolean1"":""true"",""TestNotBoolean2"":
""false"",""TestNullBoolean"":null,""TestNullDate"":null,""TestNullInteger"":0,""Test
NullString"":null,""TestNullTimestamp"":null,""TestRawJson"":{""data"":{""TestAllAsci
iChars"":"" !\""#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUV"_$c(142,143,144,
145,146,147,148,149,150,151,152,153,154,157,158,159)_" ¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿
ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ"",""TestArrayOfBoole
an"":{""ALPHA"":true,""CHARLIE"":false,""DELTA"":null,""ECHO"":false,""FOXTROT"":true
},""TestArrayOfDate"":{""Happy-Horolog :)"":""1840-12-31"",""doB"":""2125-11-27"",""n
ully"":null},""TestArrayOfInteger"":{""A THOUSAND!"":1000,""NINETY NINE"":99,""ONE"":
1,""THREE"":3,""TWO"":2,""WORLD DEBT $"":59546526326827,""WORLD WORTH $"":24100000000
0000,""nully"":null},""TestArrayOfObject"":{},""TestArrayOfString"":{""City"":""Londo
n W1"",""Country"":""UK"",""Detective"":""Sherlock Holmes"",""District"":""Marylebone
"",""FOOBAR"":""{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}"",""Street"":""221B Baker Street"",""
b4d,Th1ng5!"":""\\\""\\\""/\""\b\t\r\f\n"",""not-nully"":""'"",""nully"":null},""TestB
ooleanNo"":false,""TestBooleanYes"":true,""TestDate"":""2016-05-22"",""TestEmptyArray
```

OfBoolean"":{},""TestEmptyArrayOfDate"":{},""TestEmptyArrayOfInteger"":{},""TestEmpty
ArrayOfString"":{},""TestEmptyListOfBoolean"":[],""TestEmptyListOfOfDate"":[],""TestE
mptyListOfOfInteger"":[],""TestEmptyListOfString"":[],""TestEscapes"":""\\\""\\\""/\"
"\b\t\r\f\n"",""TestInteger"":42,""TestListOfBoolean"":[true,false,false,null,true,fa
lse,true,true,false],""TestListOfObject"":[],""TestListOfOfDate"":[""1840-12-31"",""1
841-01-10"",""1841-04-10"",""1843-09-27"",null,""1868-05-18"",""2114-10-16"",""4578-1
1-27""],""TestListOfOfInteger"":[""0"",43564356546456,0.345,0.2,null,-464356.75675,76
567,3.141592653589793238,7653475667,-65753,45676.56,-6807.383887076754324],""TestList
OfString"":[""Sherlock Holmes"",""221B Baker Street"",""Marylebone"",""London W1"",""
UK"",null,""{[{\\}[/][{[FOO,BAR]}]{\\}[/]]}"",""\\\""\\\""/\""\b\t\r\f\n"",""0"",""1"
",""3453453453453453454500""],""TestLongNumber"":499479768050558758400000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000,""TestLongString"":""Lorem ipsum dolor sit amet, consectetur
 adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex
 ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
 cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proide
nt, sunt in culpa qui officia deserunt mollit anim id est laborum. Sed ut perspiciati
s unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam
 rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae
vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur a
ut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem seq
ui nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consect
etur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et do
lore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exerci
tationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatu
r? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil mo
lestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?"",
""TestNotBoolean1"":""true"",""TestNotBoolean2"":""false"",""TestNullBoolean"":null,"
"TestNullDate"":null,""TestNullInteger"":null,""TestNullString"":null,""TestNullTimes
tamp"":null,""TestRawJson"":"""",""TestSingleObject"":"""",""TestString"":""{[{\\}[/]
[{[\""FOO\"",\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\"",\""BAR\""]}]{\\}[/]]
}"",""TestStringNotNull"":"""",""TestStringOfJSON"":""{\""menu\"": { \""id\"": \""fil
e\"", \""value\"": \""File\"", \""popup\"": { \""menuitem\"": [{\""value\"": \""New\
"", \""onclick\"": \""CreateNewdoc()\""}, {\""value\"": \""Open\"", \""onclick\"": \"
"Opendoc()\""}, {\""value\"": \""Close\"", \""onclick\"": \""Closedoc()\""}] }}}"","
"TestTimestamp"":""1966-01-27T23:12:02"",""TestTimestampShort"":""1966-01-27T00:00:00
"",""_id"":""FOO""}},""TestSingleObject"":"""",""TestString"":""{[{\\}[/][{[\""FOO\""
,\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\""\"",\""BAR\""]}]{\\}[/]]}"",""TestSt
ringNotNull"":"""",""TestStringOfJSON"":""{\""menu\"": { \""id\"": \""file\"", \""val
ue\"": \""File\"", \""popup\"": { \""menuitem\"": [{\""value\"": \""New\"", \""oncli
ck\"": \""CreateNewdoc()\""}, {\""value\"": \""Open\"", \""onclick\"": \""Opendoc()\"
"}, {\""value\"": \""Close\"", \""onclick\"": \""Closedoc()\""}] }}}"",""TestTime"":
""01:14:04"",""TestTimestamp"":""1966-01-27T23:12:02"",""TestTimestampShort"":""1966-
01-27T00:00:00"",""TestZero"":0,""ZTestJsonMethod"":""LOREM IPSUM DOLOR SIT AMET, CON
SECTETUR ADIPISCING ELIT, SED DO EIUSMOD TEMPOR INCIDIDUNT UT LABORE ET DOLORE MAGNA
ALIQUA. UT ENIM AD MINIM VENIAM, QUIS NOSTRUD EXERCITATION ULLAMCO LABORIS NISI UT AL
IQUIP EX EA COMMODO CONSEQUAT. DUIS AUTE IRURE DOLOR IN REPREHENDERIT IN VOLUPTATE VE
LIT ESSE CILLUM DOLORE EU FUGIAT NULLA PARIATUR. EXCEPTEUR SINT OCCAECAT CUPIDATAT NO
N PROIDENT, SUNT IN CULPA QUI OFFICIA DESERUNT MOLLIT ANIM ID EST LABORUM. SED UT PER
SPICIATIS UNDE OMNIS ISTE NATUS ERROR SIT VOLUPTATEM ACCUSANTIUM DOLOREMQUE LAUDANTIU
M, TOTAM REM APERIAM, EAQUE IPSA QUAE AB ILLO INVENTORE VERITATIS ET QUASI ARCHITECTO
 BEATAE VITAE DICTA SUNT EXPLICABO. NEMO ENIM IPSAM VOLUPTATEM QUIA VOLUPTAS SIT ASPE
RNATUR AUT ODIT AUT FUGIT, SED QUIA CONSEQUUNTUR MAGNI DOLORES EOS QUI RATIONE VOLUPT
ATEM SEQUI NESCIUNT. NEQUE PORRO QUISQUAM EST, QUI DOLOREM IPSUM QUIA DOLOR SIT AMET,
 CONSECTETUR, ADIPISCI VELIT, SED QUIA NON NUMQUAM EIUS MODI TEMPORA INCIDUNT UT LABO
RE ET DOLORE MAGNAM ALIQUAM QUAERAT VOLUPTATEM. UT ENIM AD MINIMA VENIAM, QUIS NOSTRU
M EXERCITATIONEM ULLAM CORPORIS SUSCIPIT LABORIOSAM, NISI UT ALIQUID EX EA COMMODI CO
NSEQUATUR? QUIS AUTEM VEL EUM IURE REPREHENDERIT QUI IN EA VOLUPTATE VELIT ESSE QUAM

```
NIHIL MOLESTIAE CONSEQUATUR, VEL ILLUM QUI DOLOREM EUM FUGIAT QUO VOLUPTAS NULLA PARI
ATUR?"","_id"":""FOO""}"
|         TestString = "{[{\}[/][{[""FOO"","""""""""""""""""""""""""""""""","""BAR""
]}]{\}[/]]}"
|  TestStringNotNull = ""
|   TestStringOfJSON = "{""menu"": { ""id"": ""file"", ""value"": ""File"", ""popup""
: { ""menuitem"": [ {""value"": ""New"", ""onclick"": ""CreateNewdoc()""}, {""value""
: ""Open"", ""onclick"": ""Opendoc()""}, {""value"": ""Close"", ""onclick"": ""Closed
oc()""} ] }}}"
|           TestTime = 4444
|      TestTimestamp = "1966-01-27 23:12:02"
| TestTimestampShort = "1966-01-27 00:00:00"
|           TestZero = ""
|    ZTestJsonMethod = ""
|      ZTestJsonName = "FOO"
+----------------- swizzled references ---------------
|i%TestArrayOfBoolean = ""
|i%TestArrayOfBoolean("ALPHA") = 1
|i%TestArrayOfBoolean("CHARLIE") = 0
|i%TestArrayOfBoolean("DELTA") = ""
|i%TestArrayOfBoolean("ECHO") = 0
|i%TestArrayOfBoolean("FOXTROT") = 1
|r%TestArrayOfBoolean = "25@%Collection.ArrayOfDT"  <Set>
|  i%TestArrayOfDate = ""
|i%TestArrayOfDate("Happy-Horolog :)") = 0
|i%TestArrayOfDate("doB") = 104434
|i%TestArrayOfDate("nully") = ""
|  r%TestArrayOfDate = "27@%Collection.ArrayOfDT"  <Set>
|i%TestArrayOfInteger = ""
|i%TestArrayOfInteger("A THOUSAND!") = 1000
|i%TestArrayOfInteger("NINETY NINE") = 99
|i%TestArrayOfInteger("ONE") = 1
|i%TestArrayOfInteger("THREE") = 3
|i%TestArrayOfInteger("TWO") = 2
|i%TestArrayOfInteger("WORLD DEBT $") = 59546526326827
|i%TestArrayOfInteger("WORLD WORTH $") = 2410000000000000
|i%TestArrayOfInteger("nully") = ""
|r%TestArrayOfInteger = "28@%Collection.ArrayOfDT"  <Set>
|i%TestArrayOfObject = ""  <Set>
|r%TestArrayOfObject = "1@%Collection.ArrayOfObj"  <Set>
|i%TestArrayOfString = ""
|i%TestArrayOfString("City") = "London W1"
|i%TestArrayOfString("Country") = "UK"
|i%TestArrayOfString("Detective") = "Sherlock Holmes"
|i%TestArrayOfString("District") = "Marylebone"
|i%TestArrayOfString("FOOBAR") = "{[{\}[/][{[FOO,BAR]}]{\}[/]]}"
|i%TestArrayOfString("Street") = "221B Baker Street"
|i%TestArrayOfString("b4d,Th1ng5!") = "\""\""/"""_$c(8,9,10,12,13)
|i%TestArrayOfString("not-nully") = ""
|i%TestArrayOfString("nully") = $c(0)
|r%TestArrayOfString = "26@%Collection.ArrayOfDT"  <Set>
|i%TestEmptyArrayOfBoolean = ""  <Set>
|r%TestEmptyArrayOfBoolean = "2@%Collection.ArrayOfDT"  <Set>
|i%TestEmptyArrayOfDate = ""  <Set>
|r%TestEmptyArrayOfDate = "3@%Collection.ArrayOfDT"  <Set>
|i%TestEmptyArrayOfInteger = ""  <Set>
|r%TestEmptyArrayOfInteger = "4@%Collection.ArrayOfDT"  <Set>
|i%TestEmptyArrayOfString = ""  <Set>
|r%TestEmptyArrayOfString = "5@%Collection.ArrayOfDT"  <Set>
```

```
|i%TestEmptyListOfBoolean = ""  <Set>
|r%TestEmptyListOfBoolean = "6@%Collection.ListOfDT"  <Set>
|i%TestEmptyListOfOfDate = ""  <Set>
|r%TestEmptyListOfOfDate = "7@%Collection.ListOfDT"  <Set>
|i%TestEmptyListOfOfInteger = ""  <Set>
|r%TestEmptyListOfOfInteger = "8@%Collection.ListOfDT"  <Set>
|i%TestEmptyListOfString = ""  <Set>
|r%TestEmptyListOfString = "9@%Collection.ListOfDT"  <Set>
|i%TestListOfBoolean = ""
|i%TestListOfBoolean(1) = 1
|i%TestListOfBoolean(2) = 0
|i%TestListOfBoolean(3) = 0
|i%TestListOfBoolean(4) = ""
|i%TestListOfBoolean(5) = 1
|i%TestListOfBoolean(6) = 0
|i%TestListOfBoolean(7) = 1
|i%TestListOfBoolean(8) = 1
|i%TestListOfBoolean(9) = 0
|r%TestListOfBoolean = "22@%Collection.ListOfDT"  <Set>
| i%TestListOfObject = ""  <Set>
| r%TestListOfObject = "10@%Collection.ListOfObj"  <Set>
| i%TestListOfOfDate = ""
|i%TestListOfOfDate(1) = 0
|i%TestListOfOfDate(2) = 10
|i%TestListOfOfDate(3) = 100
|i%TestListOfOfDate(4) = 1000
|i%TestListOfOfDate(5) = ""
|i%TestListOfOfDate(6) = 10000
|i%TestListOfOfDate(7) = 100000
|i%TestListOfOfDate(8) = 1000000
| r%TestListOfOfDate = "23@%Collection.ListOfDT"  <Set>
|i%TestListOfOfInteger = ""
|i%TestListOfOfInteger(1) = 0
|i%TestListOfOfInteger(2) = 43564356546456
|i%TestListOfOfInteger(3) = .345
|i%TestListOfOfInteger(4) = .2
|i%TestListOfOfInteger(5) = ""
|i%TestListOfOfInteger(6) = -464356.75675
|i%TestListOfOfInteger(7) = 76567
|i%TestListOfOfInteger(8) = 3.141592653589793238
|i%TestListOfOfInteger(9) = 7653475667
|i%TestListOfOfInteger(10) = -65753
|i%TestListOfOfInteger(11) = 45676.56
|i%TestListOfOfInteger(12) = -6807.383887076754324
|r%TestListOfOfInteger = "24@%Collection.ListOfDT"  <Set>
| i%TestListOfString = ""
|i%TestListOfString(1) = "Sherlock Holmes"
|i%TestListOfString(2) = "221B Baker Street"
|i%TestListOfString(3) = "Marylebone"
|i%TestListOfString(4) = "London W1"
|i%TestListOfString(5) = "UK"
|i%TestListOfString(6) = $c(0)
|i%TestListOfString(7) = "{[{\}[/][{[FOO,BAR]}]{\}[/]]}"
|i%TestListOfString(8) = "\""\""/"""_$c(8,9,10,12,13)
|i%TestListOfString(9) = 0
|i%TestListOfString(10) = 1
|i%TestListOfString(11) = 345345345345345345400
| r%TestListOfString = "21@%Collection.ListOfDT"  <Set>
| i%TestSingleObject = ""
```

```
| r%TestSingleObject = ""
+----------------------------------------------------
```

Btw, the main part of this post was written in the markdown editor and cut and paste directly into this post with syntax highlighting for COS.

Sean.

#JSON #Object Data Model #Caché

Source URL:https://community.intersystems.com/post/cogs-library