24.6.9.0 JSON und JSONCollection

Auf der Website http://de.wikipedia.org/wiki/JavaScript_Object_Notation wird das Daten-Format JSON so beschrieben: *Die JavaScript Object Notation, kurz JSON ['dʒeɪsən], ist ein kompaktes Datenformat in einer einfach lesbaren Textform zum Zweck des Datenaustauschs zwischen Anwendungen. Jedes gültige JSON-Dokument soll ein gültiges JavaScript sein und per eval() interpretiert werden können. […] Davon abgesehen ist JSON aber unabhängig von der Programmiersprache.*

- Unter https://tools.ietf.org/html/rfc7159 können Sie die RFC für JSON nachlesen.
- Das deutsche JSON-Original finden Sie unter http://json.org/json-de.html, auf das die Gambas-Dokumentation verweist.

Die Klasse JSON (gb.web → gb.util.web) erlaubt Ihnen das Kodieren und Dekodieren im JSON-Format.

24.6.9.0.1 Zuordnung JSON-Element und Gambas-Datentyp

Die folgende Tabelle beschreibt die Zuordnung zwischen JSON-Elementen und Gambas-Datentypen:

JSON-Syntax	JSON-Element	Gambas-Datentyp
{ "string": value , ... }	Objekt	Collection
[value, ...]	Array	Variant[ ]
"abc"	Zeichenkette	UTF-8-String
-123.45E+6	Zahl	Integer, Long oder Float
true oder false	Wahrheitswert (Boolean)	Boolean
null	Null	NULL oder JSON.Null

Tabelle 24.6.9.0.1 : Zuordnungstabelle

Beachten Sie, dass Sie nach der JSON-Syntax die Werte *true*, *false* und *null* im JSON-Text klein schreiben müssen.

24.6.9.0.2 Klasse JSON

Die Klasse JSON besitzt nur eine Eigenschaft und zwei Methoden:

- Die statische Eigenschaft *Null* vom Typ Variant repräsentiert einen JSON-Nullwert. Sie wird u.a. verwendet, wenn das optionale Argument 'UseNull' in der Methode JSON.Decode(...) den Wert 'True' hat.

- Die Methode *Encode (Value As Variant) As String* kodiert einen Gambas-Wert in einen JSON-Text und gibt diesen zurück. Ein Datum wird in einen entsprechenden String konvertiert, da JSON keinen Typ 'Datum' unterstützt. Beachten Sie: Der Parameter 'Value' ist in der Gambas-Terminologie ein typisierter Wert. Auch wenn Sie Value als Variant übergeben, findet der Interpreter den korrekten Datentyp von Value heraus. Das Datenaustauschformat JSON gestattet es in Gambas, vorliegende Daten mit der Funktion Encode(...) zu serialisieren. Die serialisierten Daten liegen dann als JSON-Objekt (JSON-Text) vor, dessen Struktur mit {"KEY_1": WERT_1, "KEY_2": WERT_2, ..., "KEY_k": WERT_k} einer Gambas-Collection ähnlich ist.

- Die Methode *Decode (JSONString As String [, UseNull As Boolean]) As Variant* decodiert einen JSON-Text und gibt ihn als Gambas-Wert vom Daten-Typ Variant zurück – entweder vom Typ Collection oder JSONCollection. Wenn das optionale Argument 'UseNull' in der Methode JSON.Decode den Wert 'True' hat, dann wird Null von der JSON.Null-Eigenschaft dargestellt. Das ermöglicht die Unterscheidung zwischen einer nicht festgelegten Eigenschaft und einer Eigenschaft mit dem Wert NULL.
  Die Methode Decode(...) realisiert sowohl einen Validator als auch einen Parser für einen JSON-Text. Aus Daten im JSON-Format werden Daten vom Gambas-Typ Variant, wenn der JSON-Text vom Validator als syntaktisch korrekt erkannt worden ist.

24.6.9.0.3 Klasse JSONCollection

Die Klasse JSONCollection (gb.web → gb.util.web) repräsentiert ein JSON-Objekt.

- JSONCollection ist eine spezielle Collection, die es erlaubt, einen Null-Wert mit einem Schlüssel zu assoziieren, was bei einer Collection in Gambas so nicht möglich wäre. Wenn bei einer Collection (gb) der Wert zu einem definierten Schlüssel Null ist, dann wird das Schlüssel-Wert-Paar aus der Collection entfernt!
- Die Klasse JSONCollection wird u.a von der Methode JSON.Decode(JSONString As String [, UseNull As Boolean]) verwendet, wenn der Wert für das optionale Argument 'UseNull' auf True gesetzt wurde.

Die Klasse *JSONCollection* verfügt über zwei Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Count	Integer	Gibt die Anzahl der Elemente zurück, die in der JSON-Collection gespeichert sind.
Length	Integer	Synonym für die Eigenschaft Count.
Key	String	Gibt den Schlüssel (Key) des *zuletzt* gelesenen oder aufgezählten Elements in einer JSON-Collection zurück.

Tabelle 24.6.9.0.2 : Eigenschaften der Klasse JSONCollection

Die Klasse *JSONCollection* besitzt die folgenden fünf Methoden:

Methode	Beschreibung
Add (Value As Variant, Key As String)	Fügt ein Element als Wert-Schlüssel-Paar in eine JSON-Collection ein.
Clear ( )	Löscht den Inhalt einer JSON-Collection.
Copy ( )	Gibt eine 1:1-Kopie einer JSON-Collection als eigenständiges Objekt zurück.
Exist (Key As String) As Boolean	Gibt True zurück, wenn zu dem als Parameter 'Key' übergebenen Schlüssel ein Wert in einer JSON-Collection existiert.
Remove (Key As String)	Löscht das Element mit dem als Parameter 'Key' übergebenen Schlüssel in einer JSON-Collection.

Tabelle 24.6.9.0.3 : Methoden der Klasse JSONCollection

24.6.9.0.4 Beispiel

In diesem Beispiel werden Daten mit Hilfe der Methode JSON.Decode(....) in das JSON-Format konvertiert:

```
[1]   Dim fRatio As Float, sJSONText As String, vArray As Variant[]
[2]   Dim cCollection As Collection, cJSONCollection As JSONCollection
[3]
[4]   cJSONCollection = New JSONCollection
[5]   cCollection = New Collection
[6]
[7]   fRatio = 0.1
[8]   cJSONCollection["Seite \"a\" "] = 688.7 * fRatio
[9]   cCollection[String.Chr(946)] = 43
[10]  cCollection[String.Chr(947)] = 2.039E1
[11]  cJSONCollection["Winkel"] = cCollection ' Alternative: cData.Add(cCollection, "Winkel")
[12]  cJSONCollection["Umfang Dreieck " & String.Chr(916) & " ABC"] = "?"
[13]  cJSONCollection["Widerstand in " & String.Chr(937)] = 2550
[14]  vArray = New Variant[]
[15]  vArray = ["gambas-buch.de", "gambas.sourceforge" & ".net"]
[16]  cJSONCollection["URLs"] = vArray
[17]  cJSONCollection["EMail"] = "wer@ist.da"
[18]  cJSONCollection["Datum"] = Format(Now(), "dd. mmmm yyyy")
[19]  cJSONCollection["Diplom"] = False
[20]  cJSONCollection["Master"] = JSON.Null
[21]
[22]  sJSONText = JSON.Encode(cJSONCollection)
[23]
[24]  Print sJSONText
```

Kommentar:

- In der Zeile 8 wird das erste Element in die JSON-Colletion eingefügt. Der Schlüssel enthält "a" und daher müssen die beiden " um a maskiert werden. Der Wert wird erst berechnet und dann zugewiesen.
- Das Schlüssel-Wert-Paar "Winkel" | Wert in der Zeile 11 enthält als Wert eine Collection, deren zwei Wert-Schlüssel-Paare in den Zeilen 9 und 10 festgelegt werden. Die Zeichen für Beta und Gamma erreichen Sie auf der Tastatur i.A. nicht. Deshalb werden diese utf8-codiert.
- Dem Schlüssel "URLs" wird in der Zeile 16 als Wert ein Variant-Array zugewiesen, das in der Zeile 14 deklariert wird und in der Zeile 15 ein Inline-Array zugewiesen bekommt.
- Da JSON kein eigenes Datum-Format kennt, wird ein geeignet formatierter *Datum-String* in der Zeile 18 zugewiesen.
- Das Zuweisen eines Wahrheitswertes zum Schlüssel "Diplom" erfolgt in der Zeile 19.
- Im Gegensatz zum Schlüssel "Diplom" ist der Wert für den Schlüssel "Master" nicht bekannt. Deshalb ist als Wert JSON.Null der richtige Wert. Das setzt aber voraus, dass Sie die Variable cJSONCollection mit dem Daten-Typ *JSONCollection* deklariert haben!
- Der Methode Encode(Value As Variant) wird in der Zeile 22 als Parameter die JSONCollection übergeben und der Funktionswert der Variable sJSONText zugewiesen.
- In der Zeile 24 wird der JSON-Text ausgegeben.

So präsentiert sich der JSON-Text in der Konsole der Gambas-IDE:

```
{"Seite \"a\" ": 68.87, "Winkel": {"β": 43, "γ": 20.39}, "Umfang Dreieck Δ ABC": "?", "Widerstand in Ω":
2550, "URLs": ["gambas-buch.de", "gambas.sourceforge.net"], "EMail": "wer@ist.da", "Datum": "17. April
2016", "Diplom": false, "Master": null}
```

Gibt man den JSON-Text formatiert aus → Kapitel 24.6.9.1 Projekt Formatierer für JSON-Text, dann ist er besser lesbar:

```
{
  "Seite "a" ": 68,87,
  "Winkel": {
    "β": 43,
    "γ": 20,39
  },
  "Umfang Dreieck Δ ABC": "?",
  "Widerstand in Ω": 2550,
  "URLs": [
    "gambas-buch.de",
    "gambas.sourceforge.net"
  ],
  "EMail": "wer@ist.da",
  "Datum": "17. April 2016",
  "Diplom": false,
  "Master": null
}
```

Analysiert man den Wert-Datentyp der Elemente in der JSON-Collection, dann zeigt sich die folgende Übersicht:

```
Anzahl der Elemente in der JSON-Collection: 9
-------------------------------------------------------------------------------------------------------
Schlüssel 1 : "Seite "a" " ---> Wert-Typ: Float
Schlüssel 2 : "Winkel" ---> Wert-Typ: JSONCollection
Schlüssel 3 : "Umfang Dreieck Δ ABC" ---> Wert-Typ: String
Schlüssel 4 : "Widerstand in Ω" ---> Wert-Typ: Integer
Schlüssel 5 : "URL" ---> Wert-Typ: Variant[]
Schlüssel 6 : "EMail" ---> Wert-Typ: String
Schlüssel 7 : "Datum" ---> Wert-Typ: String
Schlüssel 8 : "Diplom" ---> Wert-Typ: Boolean
Schlüssel 9 : "Master" ---> Wert-Typ: NULL oder JSON-Null
```

Hinweis: In der Revision 7744 von Gambas3 ist ein Fehler in der Klasse JSONCollection korrigiert worden, der die Eigenschaft JSON.Null nicht korrekt übernahm.