XProfan unterstützt seit X4 das Datenaustausch-Format JSON! *V0.8ß Stand 18-12-2017_01:30*

js: Integer - Handle zu einem Json-Objekt oder einer Json-Liste; **filename**: string - Dateiname (ggf. mit Pfad)
name: string - Name eines Json-Attributes (mögliche Attribute: String, Number, Boolean, Null, Object, List)
string: string - beliebiger String-Wert **wert**: integer/double - beliebiger numerischer Wert
obj - integer: Integer - Handle zu einem Json-Objekt **lst** - integer: Handle zu einer Json-Liste

js = Create("JSON"[, filename]) Ohne filename wird ein leeres Json-Objekt angelegt, mit filename wird das Objekt aus der Datei, die ein Json-Objekrt als String entrhalten muss, eingelesen.

json = Json("TEXT", js, mode)
Das Json-Objekt js wird als Json-Sting ausgegegeben. Mode 0: als eine Zeile / mode 1: in besser lesbarer Form

Json("ADDSTRING", js, [name,] string) Dem Objekt oder der List js wird ein String als Attribut hinzugefügt. Im Falle einer Liste wird kein Attributsname benötigt.

Json("ADDNUMBER", js, [name,] wert) Dem Objekt oder der List js wird ein numerischer Wert als Attribut hinzugefügt. Im Falle einer Liste wird kein Attributsname benötigt.

Json("ADDBOOL", js, [name,] wert)
Dem Objekt oder der List js wird ein boolscher Wert als Attribut hinzugefügt. Im Falle einer Liste wird kein Attributsname benötigt. Da XProfan keinen passenden Datentyp kennt steht der Wert 0 für false und 1 für true.

Json("ADDOBJECT", js, [name,] obj) Dem Objekt oder der List js wird ein Json-Objekt als Attribut hinzugefügt. Im Falle einer Liste wird kein Attributsname benötigt.

Json("ADDLIST", js, name, lst) Dem Objekt js wird eine JSon-Liste hinzugefügt. Einer Liste kann keine Liste als direktes Child hinzugefügt werden.

string = Json("GETSTRING", js, [name|idx]) Ein String wird aus dem Objekt oder der Liste js gelesen. Beim Objekt wird der Attributsname benötigt, bei der Liste der Index in der Liste.

wert = Json("GETNUMBER", js, [name|idx]) Ein numerischer Wert wird aus dem Objekt oder der Liste js gelesen. Beim Objekt wird der Attributsname benötigt, bei der Liste der Index in der Liste.

integer = Json("GETBOOL", js, [name|idx]) Ein boolscher Wert wird aus dem Objekt oder der Liste js gelesen. Beim Objekt wird der Attributsname benötigt, bei der Liste der Index in der Liste. Ergebnis: 0 oder 1.

js = Json("GETOBJECT", js, [name|idx]) Ein Objekt wird aus dem Objekt oder der Liste js gelesen. Beim Objekt wird der Attributsname benötigt, bei der Liste der Index in der Liste.

js = Json("GETLIST", js, name) Ein Liste wird aus dem Objekt js gelesen.

js = Json("NEWLIST", name) Eine leere JSon-Liste wird erzeugt.

Json("WRITE", js, filename) Das Json-Objekt js wird als Json-Sting in eine Datei geschrieben. Ideal für den Datenaustausch mit anderen Programmen

ftp("ListDir", Datei[, modus]) Wenn Modus weggelassen wird oder 0 ist, wird wie bisher ein ausführliches Listing in die Datei geschriebn, ist der Modus <> 0, stehen in der Datei nur die Dateinamen, die etwa mit move("FileToList", Datei) in die Listboxliste geschrieben werden können.

HashArray[ ] = Json("HASH", js) Die JSON-Datei wird in ein Hash-Array eingelesen! Boolsche Werte haben den "*TRUE*" oder "*FALSE*". Eine Liste hat Wert "[n]", Objekt "{n}", wobei n das Handle ist.

integer = Json("COUNT", js)
Die Anzahl der Elemente bzw. Attribute einer Liste oder eines Objektes wird ausgegeben.

string = Json("NAME", js, idx) Der Name des Elements mit dem Index idx. Der Index beginnt bei 0. Ist das Objekt eine Liste bei der die Elemente keine Namen haben, wird "[]" zurückgegeben.

string = Json("VALUE", js, idx)
Der Inhalt/Wert des Elements mit dem Index idx als String.

typ = Json("TYP", js, idx) Der Typ des Elements mit dem Index idx als String.
Typ ist ein Integer. Hier bedeuten: 1=Number / 2=String / 3=Boolean / 4=null / 5=Liste / 6=Objekt.

Hinweis: Beim Lesen aus einer Datei (CREATE) erwartet Json diese im **UTF8-Format**, ebenso wird diese bei WRITE in diesem Format geschrieben. Beim SQL-Export wird dies auch berücksichtigt.
Innerhalb von XProfan wird das gewohnte **ANSI**-Format verwendet.