File created: 13-May-2024 22:37:13 **{WMEDLEY}<lispusers>JSON.;36**

edit by: rmk

changes to: (FNS JSON-GET)

previous date: 13-May-2024 19:23:02 {WMEDLEY}<lispusers>JSON.;33

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ JSONCOMS ((FNS JSON-PARSE JSON-VALUE JSON-SKIP JSON-STRING JSON-ARRAY JSON-OBJECT JSON-AVPAIR
                      JSON-NUMBER JSON-ATOM JSON-GET)
                 (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS JSBIN JSPEEK JSBINC JSPEEKC))))

(DEFINEQ
```

(JSON-PARSE

```
  [LAMBDA (JSONSTRING)                                    ; Edited 30-Apr-2024 08:45 by rmk
```

;; Parses a JSONSTRING into a list structure. Arrays are heading with the atom ARRAY, attribute-value lists are headed by OBJECT, each then ;; followed by elements.

```
    (JSON-VALUE (CONCAT JSONSTRING])
```

(JSON-VALUE

```
  [LAMBDA (STR)                                           ; Edited 30-Apr-2024 13:48 by rmk
    (SELCHARQ (JSON-SKIP STR)
          (%[ (JSON-ARRAY STR))
          ({ (JSON-OBJECT STR))
          (%" (JSON-STRING STR))
          ((t f n)
             (JSON-ATOM STR))
          (NIL NIL)
          (JSON-NUMBER STR])
```

(JSON-SKIP

```
  [LAMBDA (STR)                                           ; Edited 30-Apr-2024 08:56 by rmk
    (bind C while (MEMB (SETQ C (JSPEEK STR))
                        (CHARCODE (SPACE CR LF TAB)))
       do (JSBIN STR) finally (RETURN C])
```

(JSON-STRING

```
  [LAMBDA (STR)                                           ; Edited 30-Apr-2024 14:39 by rmk
    (CL:WHEN (EQ (CHARCODE %")
                 (JSON-SKIP STR))
          (JSBIN STR)
          (CONCATCODES (bind C eachtime (SETQ C (JSBIN STR)) until (EQ C (CHARCODE %"))
                          collect (CL:WHEN (EQ C (CHARCODE \))
                                           (SETQ C (JSBIN STR))
                                           (CL:UNLESS (MEMB C (CHARCODE (%" \ / BACKSPACE FORM LF CR TAB u)))
                                               ;; Not checking for Hex digits after u.
                                               (ERROR "UNEXPECTED \ ESCAPE IN JSON STRING" STR)))
                                  C)))])
```

(JSON-ARRAY

```
  [LAMBDA (STR)                                           ; Edited 30-Apr-2024 13:43 by rmk
    (CL:WHEN (EQ (CHARCODE %[)
                 (JSON-SKIP STR))
          (JSBIN STR)
          [CONS 'ARRAY (if (EQ (CHARCODE %])
                           (JSON-SKIP STR))
                       then (JSBIN STR)                   ; empty
                            NIL
                       else (collect (JSON-VALUE STR) repeatuntil (SELCHARQ (JSON-SKIP STR)
                                                                       (%, (JSBIN STR)
                                                                           NIL)
                                                                       (%] (JSBIN STR)
                                                                           T)
                                                                       (ERROR "NOT A VALID JSON ARRAY" STR])])
```

(JSON-OBJECT

```
  [LAMBDA (STR)                                           ; Edited 30-Apr-2024 13:41 by rmk
```

;; Returns NIL if STR does not start with { and thus does not indicate a JSON object, a list of attribute value pairs enclosed in { }. The attributes ;; are strings separated from the values by :. The pairs are separated by commas. We return atomic attributes.

;; The empty

```
    (CL:WHEN (EQ (CHARCODE {)
                 (JSON-SKIP STR))
```

```
        (JSBIN STR)
        [CONS 'OBJECT (if (EQ (CHARCODE }) 
                              (JSON-SKIP STR))
                          then (JSBIN STR)                          ; empty
                               NIL
                         else (collect (JSON-AVPAIR STR) repeatuntil (SELCHARQ (JSON-SKIP STR)
                                                                          (%, (JSBIN STR)
                                                                              NIL)
                                                                          (} (JSBIN STR)
                                                                             T)
                                                                          (ERROR "NOT A VALID JSON OBJECT" STR])])
```

(JSON-AVPAIR

```
  [LAMBDA (STR)                                                     ; Edited 30-Apr-2024 13:31 by rmk
    (LET (A V)
         (JSON-SKIP STR)
         (SETQ A (MKATOM (JSON-STRING STR)))
         (CL:UNLESS (EQ (CHARCODE %:)
                        (JSON-SKIP STR))
             (ERROR (ERROR "NOT A VALID JSON OBJECT" STR)))
         (JSBIN STR)
         (SETQ V (JSON-VALUE STR))
         (LIST A V])
```

(JSON-NUMBER

```
  [LAMBDA (STR)                                                     ; Edited 30-Apr-2024 13:31 by rmk
    ;; Collect the characters in reverse
    (JSON-SKIP STR)
    (LET ([SIGN (CAR (MEMB (JSPEEKC STR)
                           '(+ -]
          VAL)
         (CL:WHEN SIGN
             (JSBIN STR)
             (PUSH VAL SIGN))
         (CL:WHEN (FIXP (JSPEEKC STR))
             (bind C eachtime (SETQ C (JSPEEKC STR)) while (FIXP C) do (PUSH VAL C)
                                                                       (JSBIN STR))
             (CL:WHEN (EQ (CHARCODE %.)
                          (JSPEEK STR))
                 (JSBIN STR)
                 (PUSH VAL '%.)
                 (bind C eachtime (SETQ C (JSPEEK STR)) while (FIXP C) do (PUSH VAL C)
                                                                          (JSBIN STR)))
             (CL:WHEN (MEMB (JSPEEK STR)
                            (CHARCODE (E e)))
                 (JSBIN STR)
                 (PUSH VAL 'E)
                 (CL:WHEN [SETQ SIGN (MEMB (JSPEEK STR)
                                           '(+ -]
                     (JSBIN STR)
                     (PUSH VAL SIGN))
                 (bind C eachtime (SETQ C (JSPEEK STR)) while (FIXP C) do (PUSH VAL C)
                                                                          (JSBIN STR)))
             (PACK (DREVERSE VAL)))])
```

(JSON-ATOM

```
  [LAMBDA (STR)                                                     ; Edited 30-Apr-2024 13:31 by rmk
    (JSON-SKIP STR)
    (SELCHARQ (JSPEEK STR)
         (t (JSBIN STR)
            (if (AND (EQ (CHARCODE r)
                         (JSBIN STR))
                     (EQ (CHARCODE u)
                         (JSBIN STR))
                     (EQ (CHARCODE e)
                         (JSBIN STR)))
                then 'true
              else (ERROR "INVALID JSON STRING" STR)))
         (f (JSBIN STR)
            (if (AND (EQ (CHARCODE a)
                         (JSBIN STR))
                     (EQ (CHARCODE l)
                         (JSBIN STR))
                     (EQ (CHARCODE s)
                         (JSBIN STR))
                     (EQ (CHARCODE e)
                         (JSBIN STR)))
                then 'false
              else (ERROR "INVALID JSON STRING" STR)))
         (n (JSBIN STR)
            (if (AND (EQ (CHARCODE u)
                         (JSBIN STR))
                     (EQ (CHARCODE l)
```

```
                        (JSBIN STR))
                    (EQ (CHARCODE l)
                        (JSBIN STR)))
                then 'null
               else (ERROR "INVALID JSON STRING" STR)))
          NIL])
```

(JSON-GET

```
  [LAMBDA (OBJECT ATTRIBUTES)                                    ; Edited 13-May-2024 22:35 by rmk
                                                                 ; Edited 30-Apr-2024 14:26 by rmk
```

;; Returns the value at the end of a chain of ATTRIBUTES in OBJECT

```
    (for A (OBJ _ OBJECT) inside ATTRIBUTES do (if (EQ 'OBJECT (CAR (LISTP OBJ)))
                                                   then [SETQ OBJ (CADR (ASSOC A (CDR OBJ]
                                                 else (RETURN NIL))
       finally (RETURN OBJ])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS JSBIN MACRO (= . GNCCODE))

(PUTPROPS JSPEEK MACRO (= . CHCON1))

(PUTPROPS JSBINC MACRO ((STR)
                        (CHARACTER (JSBIN STR))))

(PUTPROPS JSPEEKC MACRO ((STR)
                         (CHARACTER (JSPEEK STR))))
)
)
```

FUNCTION INDEX

MACRO INDEX