File created: 27-Jul-90 07:36:13 **{DSK}<usr>local>lde>loops>src>SYSTEM>BLOCKLOOKUP.;2**

changes to: (VARS BLOCKLOOKUPCOMS)

previous date: 11-Jun-86 16:56:14 {DSK}<usr>local>lde>loops>src>SYSTEM>BLOCKLOOKUP.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ BLOCKLOOKUPCOMS
       ((VARS \BlockIncrement \InitBlockSize)
        (GLOBALVARS \BlockIncrement \InitBlockSize)
        (MACROS \FindEntryIndex \GetNthEntry \PutNthEntry \WordFromPtrIndex)
        (FNS \AddBlockEntry \BlockFromList \BlockFull \ClearBlock \DeleteNthEntry \FindEntryIndex
             \FreeEntryIndex \GetNthEntry \GrowBlock \ListFromBlock \MakeBlock \NextBlockSize \PrintBlock
             \PutNthEntry)))

(RPAQQ \BlockIncrement 8)

(RPAQQ \InitBlockSize 8)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \BlockIncrement \InitBlockSize)
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \FindEntryIndex MACRO [OPENLAMBDA (KEY BLOCK)
                                  (DECLARE (LOCALVARS . T))
                                  (AND BLOCK (for I from 0 by 2 bind val do (COND
                                                                              ((NULL (SETQ val (\GETBASEPTR BLOCK
                                                                                                            I)))
                                                                               (* End is marked by NIL)
                                                                                (RETURN NIL))
                                                                              ((EQ KEY val)
                                                                               (RETURN I])

(PUTPROPS \GetNthEntry MACRO ((BLOCK WORDNUMBER)                              (* WORDNUMBER is used for position rather than the entry number)
                              (\GETBASEPTR BLOCK WORDNUMBER)))

(PUTPROPS \PutNthEntry MACRO ((BLOCK NTHWORD VAL)                             (* Store VAL in position NTHWORD)
                              (\RPLPTR BLOCK NTHWORD VAL)))

(PUTPROPS \WordFromPtrIndex MACRO ((ptrIndex)
                                   (LLSH ptrIndex 1)))
)

(DEFINEQ
```

(\AddBlockEntry

```
  [LAMBDA (BLOCK ENTRY freeIndex)                                             (* dgb%: "18-OCT-83 17:32")
```

(* Add an entry to this block. If it is full, then create a new block with same contents, and add entry to new block. Return block containing new entry)

```
    (OR freeIndex (SETQ freeIndex (\FreeEntryIndex BLOCK)))
    [COND
       ((NULL BLOCK)
        (SETQ BLOCK (\MakeBlock)))
       ((\BlockFull BLOCK freeIndex)
```

(* New block has free index in same place, since there are the same number of entries)

```
        (SETQ BLOCK (\GrowBlock BLOCK freeIndex]
    (\PUTBASEPTR BLOCK (IPLUS freeIndex 2)
          NIL)
    (\RPLPTR BLOCK freeIndex ENTRY)
    BLOCK])
```

(\BlockFromList

```
  [LAMBDA (lst extractFn block)                                               (* dgb%: "25-MAY-83 16:00")
```

(* Fill in the block from a list. Use extractFn to obtain part of lst which is relevant -- NIL means use it all.)

```
    [COND
       (block (\ClearBlock block))
```

```
      (T (SETQ block (\MakeBlock (LENGTH lst]
   [for item in lst do (\AddBlockEntry block (COND
                                          (extractFn (APPLY* extractFn item))
                                          (T item]
   block])
```

(\BlockFull

```
[LAMBDA (BLOCK freeIndex)                                   (* dgb%: "24-MAY-83 05:39")
                                                            (* Block is full if pointer after freeIndex contains a T)
   (OR freeIndex (SETQ freeIndex (\FreeEntryIndex BLOCK)))
   (EQ T (\GETBASEPTR BLOCK (IPLUS freeIndex 2])
```

(\ClearBlock

```
[LAMBDA (block)                                             (* dgb%: "25-MAY-83 11:06")

        (* Clear the block up to the first free entry in the block. It will contain a NIL.)

   (for i from 0 by 2 do (COND
                           ((NULL (\GETBASEPTR block i))
                            (RETURN block))
                           (T (\RPLPTR block i NIL])
```

(\DeleteNthEntry

```
[LAMBDA (BLOCK N freePos)                                   (* dgb%: "25-Jun-84 16:48")

        (* Delete the entry at word position N by moving the last one to position N, unless N is the last position freePos is sent in
        because there might be a NIL too early in the BLOCK in some cases)

   (PROG ((lastPos (IDIFFERENCE (OR freePos (\FreeEntryIndex BLOCK))
                          2)))
         [COND
            ((NEQ N lastPos)
             (\RPLPTR BLOCK N (\GETBASEPTR BLOCK lastPos]
         (\RPLPTR BLOCK lastPos NIL)
         (RETURN BLOCK])
```

(\FindEntryIndex

```
[LAMBDA (KEY BLOCK)                                         (* dgb%: "24-MAY-83 05:56")
                                                            (* Search BLOCK for KEY, returning its index, or NIL on failure)
   (DECLARE (LOCALVARS . T))
   (AND BLOCK (for I from 0 by 2 bind val do (COND
                                          ((NULL (SETQ val (\GETBASEPTR BLOCK I)))
                                                            (* End is marked by NIL)
                                           (RETURN NIL))
                                          ((EQ KEY val)
                                           (RETURN I])
```

(\FreeEntryIndex

```
[LAMBDA (block)                                             (* dgb%: "18-OCT-83 15:56")

        (* Find the index of the first free entry in the block. It will contain a NIL.
        It may not be usable, so it must be checked by \BlockFull. Index is a word pointer)

   (COND
      [block (for i from 0 by 2 do (COND
                                     ((NULL (\GETBASEPTR block i))
                                      (RETURN i]
      (T 0])
```

(\GetNthEntry

```
[LAMBDA (BLOCK WORDNUMBER)                                  (* dgb%: "24-MAY-83 05:58")
                                                            (* WORDNUMBER is used for postion position rather than the
                                                            entry number)
   (\GETBASEPTR BLOCK WORDNUMBER])
```

(\GrowBlock

```
[LAMBDA (BLOCK freeIndex)                                   (* edited%: " 3-Apr-86 18:00")

        (* * Copy contents of old block into new larger block. Return new block)

   (OR freeIndex (SETQ freeIndex (\FreeEntryIndex BLOCK)))
   (for I from 0 by 2 to freeIndex bind (NEWBLOCK _ (\MakeBlock (IPLUS freeIndex \BlockIncrement)))
      do (\RPLPTR NEWBLOCK I (\GETBASEPTR BLOCK I)) finally (RETURN NEWBLOCK])
```

(\ListFromBlock

```
[LAMBDA (block freeIndex)                                   (* dgb%: "24-MAY-83 06:08")
                                                            (* create a list containing contents of block, up to but not
                                                            including freeIndex)
   (OR freeIndex (SETQ freeIndex (\FreeEntryIndex block)))
```

```
    (COND
       ((OR (NULL block)
            (LISTP block))
        block)
       (T (for i from 0 by 2 to (IDIFFERENCE freeIndex 2) collect (\GetNthEntry block i])
```

(\MakeBlock

```
  [LAMBDA (numPointers)                                          (* dgb%: "18-OCT-83 17:27")

          (* Allocate a block of storage for a search table for pointers. In this implementation, search is assumed to be done linearly.
          The last cell of the table contains a T -- The first free entry contains)

    (PROG (BLOCK (size (\NextBlockSize numPointers)))
          (SETQ BLOCK (\ALLOCBLOCK size T))                      (* First free pointer in 0)
          (\PUTBASEPTR BLOCK 0 NIL)                              (* Set last cell)
          (\PUTBASEPTR BLOCK (\WordFromPtrIndex (SUB1 size))
                 T)
          (RETURN BLOCK])
```

(\NextBlockSize

```
  [LAMBDA (length)                                               (* edited%: " 3-Apr-86 18:01")
    (COND
       ((OR (NULL length)
            (IGREATERP \InitBlockSize length))
        \InitBlockSize)
       (T (IPLUS \InitBlockSize (ITIMES \BlockIncrement (ADD1 (IQUOTIENT (IDIFFERENCE length \InitBlockSize)
                                                                         \BlockIncrement])
```

(\PrintBlock

```
  [LAMBDA (BLOCK freeIndex)                                      (* dgb%: "24-MAY-83 06:04")
                                                                 (* Print out contents of BLOCK up to freeIndex)
    (OR freeIndex (SETQ freeIndex (\FreeEntryIndex BLOCK)))
    (for I from 0 by 2 to (IDIFFERENCE freeIndex 2) do (PRIN1 I)
                                                       (SPACES 1)
                                                       (PRINT (\GETBASEPTR BLOCK I])
```

(\PutNthEntry

```
  [LAMBDA (BLOCK NTHWORD VAL)                                    (* dgb%: "24-MAY-83 06:05")
                                                                 (* Store VAL in postion NTHWORD)
    (\RPLPTR BLOCK NTHWORD VAL])

)

(PUTPROPS BLOCKLOOKUP COPYRIGHT ("Venue & Xerox Corporation" 1983 1984 1985 1986 1990))
```

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX