

File created: 11-Jun-90 14:33:44 {DSK}<usr>local>lde>lispcore>library>CASH-FILE.;2

changes to: (IL:VARS IL:CASH-FILECOMS)

previous date: 9-Oct-87 11:22:19 {DSK}<usr>local>lde>lispcore>library>CASH-FILE.;1

Read Table: XCL

Package: CASH-FILE

Format: XCCS

; Copyright (c) 1987, 1990 by Venue & Xerox Corporation. All rights reserved.

```
(IL:RPAQQ IL:CASH-FILECOMS
  ((IL:P (PROVIDE "CASH-FILE")
    (EXPORT ' (MAKE-CASH-FILE OPEN-CASH-FILE GET-CASH-FILE REM-CASH-FILE CASH-FILE CASH-FILE-P
      CASH-FILE-HASH-FILE)
      "CASH-FILE")
    (REQUIRE "HASH-FILE" "HASH-FILE.DFASL")
    (USE-PACKAGE "HASH-FILE" "CASH-FILE")))
  (IL:STRUCTURES CASH-FILE)
  (IL:FUNCTIONS %PRINT-CASH-FILE)
  (IL:VARIABLES NOT-IN-HASH-FILE)
  (IL:FUNCTIONS MAKE-CASH-FILE OPEN-CASH-FILE GET-CASH-FILE PUT-CASH-FILE REM-CASH-FILE)
  (IL:SETFS GET-CASH-FILE)
  (IL:FUNCTIONS MOVE-TO-HEAD-OF-QUEUE ADD-TO-CACHE DEL-FROM-CACHE)
  (IL:PROP (IL:MAKEFILE-ENVIRONMENT IL:FILETYPE)
    IL:CASH-FILE)))

(PROVIDE "CASH-FILE")

(EXPORT ' (MAKE-CASH-FILE OPEN-CASH-FILE GET-CASH-FILE REM-CASH-FILE CASH-FILE CASH-FILE-P CASH-FILE-HASH-FILE)
  "CASH-FILE")

(REQUIRE "HASH-FILE" "HASH-FILE.DFASL")

(USE-PACKAGE "HASH-FILE" "CASH-FILE")

(DEFSTRUCT (CASH-FILE (:CONSTRUCTOR MAKE-CASH-FILE-INTERNAL)
  (:PRINT-FUNCTION %PRINT-CASH-FILE))
  (CACHE NIL :TYPE HASH-TABLE :READ-ONLY T)
  (CACHE-SIZE NIL :TYPE INTEGER :READ-ONLY T)
  (QUEUE NIL :TYPE LIST)
  (HASH-FILE NIL :TYPE HASH-FILE :READ-ONLY T))

(DEFUN %PRINT-CASH-FILE (CASH-FILE STREAM DEPTH)
  (FORMAT STREAM "#<Cash-File on ~A>" (LET* ((STREAM (HASH-FILE::HASH-FILE-STREAM (CASH-FILE-HASH-FILE
    CASH-FILE)))
    (NAMESTRING (NAMESTRING (PATHNAME STREAM))))
    (IF NAMESTRING
      NAMESTRING
      STREAM))))

(DEFCONSTANT NOT-IN-HASH-FILE ' (NOT-IN-HASH-FILE))

(DEFUN MAKE-CASH-FILE (FILE-NAME SIZE CACHE-SIZE)
  (MAKE-CASH-FILE-INTERNAL :HASH-FILE (MAKE-HASH-FILE FILE-NAME SIZE)
    :CACHE
    (MAKE-HASH-TABLE :SIZE CACHE-SIZE :TEST 'EQUAL)
    :CACHE-SIZE CACHE-SIZE))

(DEFUN OPEN-CASH-FILE (FILE-NAME CACHE-SIZE &KEY (DIRECTION :INPUT))
  (MAKE-CASH-FILE-INTERNAL :HASH-FILE (OPEN-HASH-FILE FILE-NAME :DIRECTION DIRECTION)
    :CACHE
    (MAKE-HASH-TABLE :SIZE CACHE-SIZE :TEST 'EQUAL)
    :CACHE-SIZE CACHE-SIZE))

(DEFUN GET-CASH-FILE (KEY CASH-FILE &OPTIONAL DEFAULT)
  (MULTIPLE-VALUE-BIND (VALUE FOUND?)
    (GETHASH KEY (CASH-FILE-CACHE CASH-FILE)))
  (COND
    (FOUND?
      ;; cache hit
      (MOVE-TO-HEAD-OF-QUEUE KEY CASH-FILE)
      (IF (EQ VALUE NOT-IN-HASH-FILE)
        ;; it was a cached miss
        (VALUES DEFAULT NIL)
        ;; it was a cached hit
```

```

      (VALUES
        ;; return a copy to be compatible with GET-HASH-FILE which always hands you new structure
        (COPY-TREE VALUE)
        T)))
  (T ;; try the HASH-FILE
    (MULTIPLE-VALUE-SETQ (VALUE FOUND?)
      (GET-HASH-FILE KEY (CASH-FILE-HASH-FILE CASH-FILE)))
    ;; cache what we found
    (ADD-TO-CACHE KEY (IF FOUND?
      ;; cache the VALUE
      VALUE
      ;; cache the miss
      NOT-IN-HASH-FILE)
      CASH-FILE)
    ;; return VALUE or DEFAULT
    (IF FOUND?
      (VALUES VALUE T)
      (VALUES DEFAULT NIL))))))

```

```

(DEFUN PUT-CASH-FILE (KEY CASH-FILE VALUE)
  ;; add it to the hash file
  (SETF (GET-HASH-FILE KEY (CASH-FILE-HASH-FILE CASH-FILE))
    VALUE)
  ;; add it to the cache
  (ADD-TO-CACHE KEY VALUE CASH-FILE)
  VALUE)

```

```

(DEFUN REM-CASH-FILE (KEY CASH-FILE)
  (LET ((FOUND? (REM-HASH-FILE KEY (CASH-FILE-HASH-FILE CASH-FILE))))
    (WHEN FOUND? (DEL-FROM-CACHE KEY CASH-FILE))
    FOUND?))

```

```

(DEFSETF GET-CASH-FILE PUT-CASH-FILE)

```

```

(DEFUN MOVE-TO-HEAD-OF-QUEUE (KEY CASH-FILE)
  (SETF (CASH-FILE-QUEUE CASH-FILE)
    (DELETE KEY (CASH-FILE-QUEUE CASH-FILE)
      :TEST
      'EQUAL :COUNT 1))
  (PUSH KEY (CASH-FILE-QUEUE CASH-FILE)))

```

```

(DEFUN ADD-TO-CACHE (KEY VALUE CASH-FILE)
  (LET ((CACHE (CASH-FILE-CACHE CASH-FILE))
    (IF (>= (HASH-TABLE-COUNT CACHE)
      (CASH-FILE-CACHE-SIZE CASH-FILE))
      ;; cache is full -- throw out last entry
      (DEL-FROM-CACHE (CAR (LAST (CASH-FILE-QUEUE CASH-FILE)))
        CASH-FILE))
      ;; store VALUE in the cache
      (SETF (GETHASH KEY CACHE)
        VALUE)
      ;; put the KEY at the head of the QUEUE
      (PUSH KEY (CASH-FILE-QUEUE CASH-FILE))
      VALUE))

```

```

(DEFUN DEL-FROM-CACHE (KEY CASH-FILE)
  ;; delete it from the queue
  (SETF (CASH-FILE-QUEUE CASH-FILE)
    (DELETE KEY (CASH-FILE-QUEUE CASH-FILE)
      :TEST
      'EQUAL :COUNT 1))
  ;; delete it from the cache
  (REMHASH KEY (CASH-FILE-CACHE CASH-FILE)))

```

```

(IL:PUTPROPS IL:CASH-FILE IL:MAKEFILE-ENVIRONMENT (:READTABLE "XCL" :PACKAGE (DEFPACKAGE "CASH-FILE"
  (:USE "LISP" "XCL"))))

```

```

(IL:PUTPROPS IL:CASH-FILE IL:FILETYPE :XCL-COMPILE-FILE)

```

{MEDLEY}<library>CASH-FILE.;1

Page 3

(IL:PUTPROPS **IL:CASH-FILE IL:COPYRIGHT** ("Venue & Xerox Corporation" 1987 1990))

FUNCTION INDEX

%PRINT-CASH-FILE	1	GET-CASH-FILE	1	OPEN-CASH-FILE	1
ADD-TO-CACHE	2	MAKE-CASH-FILE	1	PUT-CASH-FILE	2
DEL-FROM-CACHE	2	MOVE-TO-HEAD-OF-QUEUE	2	REM-CASH-FILE	2

PROPERTY INDEX

IL:CASH-FILE	2
--------------------	---

SETF INDEX

GET-CASH-FILE	2
---------------------	---

CONSTANT INDEX

NOT-IN-HASH-FILE	1
------------------------	---

STRUCTURE INDEX

CASH-FILE	1
-----------------	---
