

File created: 18-May-90 01:15:40 {DSK}<usr>local>lde>lispcore>sources>XCL-EXTRAS.;2

changes to: (IL:VARS IL:XCL-EXTRASCOMS)

previous date: 11-Jan-88 16:59:17 {DSK}<usr>local>lde>lispcore>sources>XCL-EXTRAS.;1

Read Table: XCL

Package: XEROX-COMMON-LISP

Format: XCCS

; Copyright (c) 1987, 1988, 1990 by Venue & Xerox Corporation. All rights reserved.

(IL:RPAQQ **IL:XCL-EXTRASCOMS**

```
(;; Finess the define-file-info problem
(IL:DEFINE-TYPES FILE-ENVIRONMENTS)
(IL:FUNCTIONS DEFINE-FILE-ENVIRONMENT)
;; Macro for writing macros
(IL:FUNCTIONS ONCE-ONLY)
;; CL veneer on IL record module
(IL:FUNCTIONS RECORD-FETCH SETF-FETCH RECORD-FFETCH SETF-FFETCH RECORD-CREATE)
(IL:SETFS RECORD-FETCH RECORD-FFETCH)
;; An alternate version of the above
(IL:FUNCTIONS DEFINE-RECORD)
(IL:FUNCTIONS RECORD-ACCESS-MACRO SETF-RECORD-ACCESS-MACRO RECORD-PREDICATE-MACRO
RECORD-CONSTRUCTOR-MACRO)
(IL:PROP (IL:FILETYPE IL:MAKEFILE-ENVIRONMENT)
IL:XCL-EXTRAS))
```

;; Finess the define-file-info problem

(DEF-DEFINE-TYPE **FILE-ENVIRONMENTS** "File info")

(DEFDEFINER **DEFINE-FILE-ENVIRONMENT** FILE-ENVIRONMENTS (FILE &KEY READTABLE PACKAGE BASE COMPILER)

```
(LET ((ROOTNAME (INTERN (STRING FILE)
(FIND-PACKAGE "INTERLISP"))))
(EVAL-WHEN (EVAL LOAD)
,@(IF (OR READTABLE PACKAGE BASE)
\((SETF (GET ',ROOTNAME 'IL:MAKEFILE-ENVIRONMENT)
',@(IF READTABLE
\(:READTABLE ,READTABLE))
,@(IF PACKAGE
\(:PACKAGE ,PACKAGE))
,@(IF BASE
\(:BASE ,BASE))))))
,@(IF COMPILER
\((SETF (GET ',ROOTNAME 'IL:FILETYPE)
',COMPILER))))))
```

;; Macro for writing macros

(DEFMACRO **ONCE-ONLY** (VARS &BODY BODY)

;;; ONCE-ONLY assures that the forms given as vars are evaluated in the proper order, once only. Used in the body of macro definitions. Taken from  
;;; Zeta Lisp.

```
(LET* ((GENSYM-VAR (GENSYM))
(RUN-TIME-VARS (GENSYM))
(RUN-TIME-VALS (GENSYM))
(EXPAND-TIME-VAL-FORMS (MAPCAR #'(LAMBDA (VAR)
\ (IF (OR (SYMBOLP ,VAR)
(CONSTANTP ,VAR))
,VAR
(LET ((,GENSYM-VAR (GENSYM))
(PUSH ,GENSYM-VAR ,RUN-TIME-VARS)
(PUSH ,VAR ,RUN-TIME-VALS)
,GENSYM-VAR)))
VARS)))
\(',LET* (,RUN-TIME-VARS ,RUN-TIME-VALS (WRAPPED-BODY (,LET
, (WITH-COLLECTION
(DO ((VAR VARS (CDR VAR))
(EXPAND-TIME-VAL-FORM
EXPAND-TIME-VAL-FORMS (CDR
EXPAND-TIME-VAL-FORM
)))
(NULL VAR))
(COLLECT (LIST (CAR VAR)
(CAR EXPAND-TIME-VAL-FORM)))
,@BODY)))
```

```

      `('LET , (WITH-COLLECTION (DO ((RUN-TIME-VAR (REVERSE ,RUN-TIME-VARS)
                                         (CDR RUN-TIME-VAR))
                                   (RUN-TIME-VAL (REVERSE ,RUN-TIME-VALS)
                                         (CDR RUN-TIME-VAL)))
                                   ((NULL RUN-TIME-VAR))
                                   (COLLECT (LIST (CAR RUN-TIME-VAR)
                                                (CAR RUN-TIME-VAL))))))
      , WRAPPED-BODY)))

```

:: CL veneer on IL record module

```

(DEFMACRO RECORD-FETCH (RECORD FIELD OBJECT)
  (LET ((IL-RECORD (INTERN (STRING RECORD)
                           IL:*INTERLISP-PACKAGE*))
        (IL-FIELD (INTERN (STRING FIELD)
                           IL:*INTERLISP-PACKAGE*)))
    `(IL:fetch | (, IL-RECORD , IL-FIELD) IL:|of| , OBJECT)))

```

```

(DEFMACRO SETF-FETCH (RECORD FIELD OBJECT NEW-VALUE)
  (LET ((IL-RECORD (INTERN (STRING RECORD)
                           IL:*INTERLISP-PACKAGE*))
        (IL-FIELD (INTERN (STRING FIELD)
                           IL:*INTERLISP-PACKAGE*)))
    `(IL:replace | (, IL-RECORD , IL-FIELD) IL:|of| , OBJECT IL:|with| , NEW-VALUE)))

```

```

(DEFMACRO RECORD-FFETCH (RECORD FIELD OBJECT)
  (LET ((IL-RECORD (INTERN (STRING RECORD)
                           IL:*INTERLISP-PACKAGE*))
        (IL-FIELD (INTERN (STRING FIELD)
                           IL:*INTERLISP-PACKAGE*)))
    `(IL:ffetch | (, IL-RECORD , IL-FIELD) IL:|of| , OBJECT)))

```

```

(DEFMACRO SETF-FFETCH (RECORD FIELD OBJECT NEW-VALUE)
  (LET ((IL-RECORD (INTERN (STRING RECORD)
                           IL:*INTERLISP-PACKAGE*))
        (IL-FIELD (INTERN (STRING FIELD)
                           IL:*INTERLISP-PACKAGE*)))
    `(IL:freplace | (, IL-RECORD , IL-FIELD) IL:|of| , OBJECT IL:|with| , NEW-VALUE)))

```

```

(DEFMACRO RECORD-CREATE (RECORD &REST KEYWORD-PAIRS)
  (LET ((IL-RECORD (INTERN (STRING RECORD)
                           IL:*INTERLISP-PACKAGE*))
        ` (IL:create | , IL-RECORD , @ (WITH-COLLECTION (DO ((KEYWORD KEYWORD-PAIRS (CDDR KEYWORD))
                                                            (VALUE (CDR KEYWORD-PAIRS)
                                                            (CDDR VALUE))
                                                            KEYWORD-SYMBOL)
                                                            (NULL KEYWORD))
                                                            (SETQ KEYWORD-SYMBOL (INTERN (STRING (CAR KEYWORD))
                                                                    IL:*INTERLISP-PACKAGE*))
                                                            (COLLECT KEYWORD-SYMBOL)
                                                            (IF (NOT (MEMBER KEYWORD-SYMBOL ' (IL:USING IL:COPYING
                                                                    IL:REUSING IL:SMASHING)
                                                                    :TEST
                                                                    #'EQ))
                                                                (COLLECT 'IL:_)
                                                                (COLLECT (CAR VALUE))))))))

```

```

(DEFSETF RECORD-FETCH SETF-FETCH)

```

```

(DEFSETF RECORD-FFETCH SETF-FFETCH)

```

:: An alternate version of the above

```

(DEFDEFINER DEFINE-RECORD IL:STRUCTURES (RECORD-NAME INTERLISP-RECORD-NAME &KEY (CONC-NAME NIL CONC-NAME-P)
                                         (CONSTRUCTOR NIL CONSTRUCTOR-P)
                                         (PREDICATE NIL PREDICATE-P)
                                         (FAST-ACCESSORS NIL)
                                         (PACKAGE *PACKAGE*))
  (IF (NOT (PACKAGEP PACKAGE))
      (SETQ PACKAGE (FIND-PACKAGE PACKAGE)))
  (SETQ CONC-NAME (IF CONC-NAME-P
                     (IF CONC-NAME
                         (STRING CONC-NAME)
                         "")
                     (CONCATENATE 'STRING (STRING RECORD-NAME)
                                   "-")))
  (LET
    ((DECLARATION (IL:RELOOK INTERLISP-RECORD-NAME))
     (FIELD-NAMES (IL:RECORDFIELDNAMES INTERLISP-RECORD-NAME)))

```

```
(IF (NULL DECLARATION)
  (ERROR "Record ~s not yet defined." INTERLISP-RECORD-NAME))
` (EVAL-WHEN (COMPILE LOAD EVAL)
  ,@(MAPCAN #'(LAMBDA (FIELD-NAME)
    (WHEN FIELD-NAME
      (LET ((NEW-NAME (INTERN (CONCATENATE 'STRING CONC-NAME (STRING FIELD-NAME))
        PACKAGE)))
        `( (SETF (MACRO-FUNCTION ',NEW-NAME)
          'RECORD-ACCESS-MACRO)
          (CL::SET-SHARED-SETF-INVERSE ',NEW-NAME 'SETF-RECORD-ACCESS-MACRO)
          (SETF (GET ',NEW-NAME :SLOT-INFO)
            ', `((,INTERLISP-RECORD-NAME ,FIELD-NAME)
              ,FAST-ACCESSORS))))))
        FIELD-NAMES)
    ,@(LET ((NEW-NAME (IF PREDICATE-P
      PREDICATE
      (INTERN (CONCATENATE 'STRING (STRING RECORD-NAME)
        "-P")
        PACKAGE))))
      (WHEN (AND NEW-NAME (OR (EQ (CAR DECLARATION)
        'IL:DATATYPE)
        (FIND-IF #'(LAMBDA (CLAUSE)
          (AND (CONSP CLAUSE)
            (OR (EQ (CAR CLAUSE)
              'IL:TYPE?)
              (EQ (CAR CLAUSE)
                'IL:|type?|))))
          DECLARATION)))
        `( (SETF (MACRO-FUNCTION ',NEW-NAME)
          'RECORD-PREDICATE-MACRO)
          (SETF (GET ',NEW-NAME :TYPE-INFO)
            ', INTERLISP-RECORD-NAME))))
      ,@(LET ((NEW-NAME (IF CONSTRUCTOR-P
        CONSTRUCTOR
        (INTERN (CONCATENATE 'STRING "MAKE-" (STRING RECORD-NAME)
          PACKAGE))))
          (WHEN (AND NEW-NAME (OR (NOT (MEMBER (CAR DECLARATION)
            ' (IL:BLOCKRECORD IL:ACCESSFNS IL:ATOMRECORD)
            :TEST
            #'EQ))
            (FIND-IF #'(LAMBDA (CLAUSE)
              (IF (CONSP CLAUSE)
                (OR (EQ (CAR CLAUSE)
                  'IL:CREATE)
                  (EQ (CAR CLAUSE)
                    'IL:|create|))
                (EQ CLAUSE INTERLISP-RECORD-NAME)))
              (CDDR DECLARATION))))
            `( (SETF (MACRO-FUNCTION ',NEW-NAME)
              'RECORD-CONSTRUCTOR-MACRO)
              (SETF (GET ',NEW-NAME :FIELD-INFO)
                ', (,INTERLISP-RECORD-NAME ,FIELD-NAMES))))))))))
```

```
(DEFUN RECORD-ACCESS-MACRO (FORM &OPTIONAL ENV)
  (DECLARE (IGNORE ENV))
  (DESTRUCTURING-BIND (SPECIFIER FAST-ACCESSOR-P)
    (OR (GET (CAR FORM)
      :SLOT-INFO)
      (ERROR "No slot information cached."))
    (IF FAST-ACCESSOR-P
      `( (IL:|ffetch| ,SPECIFIER IL:|of| , (SECOND FORM))
        '(IL:|fetch| ,SPECIFIER IL:|of| , (SECOND FORM))))))
```

```
(CL::DEFINE-SHARED-SETF-MACRO SETF-RECORD-ACCESS-MACRO ACCESSOR (DATUM) (NEW-VALUE)
  (DESTRUCTURING-BIND (SPECIFIER FAST-ACCESSOR-P)
    (OR (GET ACCESSOR :SLOT-INFO)
      (ERROR "No slot information cached."))
    (IF FAST-ACCESSOR-P
      `( (IL:|replace| ,SPECIFIER IL:|of| ,DATUM IL:|with| ,NEW-VALUE)
        '(IL:|replace| ,SPECIFIER IL:|of| ,DATUM IL:|with| ,NEW-VALUE))))))
```

```
(DEFUN RECORD-PREDICATE-MACRO (FORM &OPTIONAL ENV)
  (DECLARE (IGNORE ENV))
  `( (IL:|type?| , (OR (GET (CAR FORM)
    :TYPE-INFO)
    (ERROR "No type information cached."))
    , (SECOND FORM)))
```

```
(DEFUN RECORD-CONSTRUCTOR-MACRO (FORM &OPTIONAL ENV)
  (DECLARE (IGNORE ENV))
  (DESTRUCTURING-BIND (TYPE FIELD-NAMES)
    (OR (GET (CAR FORM)
      :FIELD-INFO)
```

```

(ERROR "No field information cached.")
` (IL:create| ,TYPE ,@(WITH-COLLECTION (DO* ((KEYWORD (CDR FORM)
(CDDR KEYWORD))
(KEYWORD-SYMBOL (CAR KEYWORD)
(CAR KEYWORD))
(VALUE (CADR KEYWORD)
(CADR KEYWORD))
RESERVED-WORD)
((NULL KEYWORD))
(SETQ RESERVED-WORD (CAR (MEMBER KEYWORD-SYMBOL
' (IL:USING IL:COPYING
IL:REUSING IL:SMASHING)
:TEST
' STRING=)))
(COLLECT (OR RESERVED-WORD (CAR (MEMBER KEYWORD-SYMBOL
FIELD-NAMES :TEST
' STRING=))))))
(IF (NOT RESERVED-WORD)
(COLLECT 'IL:_))
(COLLECT VALUE))))))

```

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

(IL:PUTPROPS IL:XCL-EXTRAS IL:MAKEFILE-ENVIRONMENT (:READTABLE "XCL" :PACKAGE "XCL"))

(IL:PUTPROPS IL:XCL-EXTRAS IL:COPYRIGHT ("Venue & Xerox Corporation" 1987 1988 1990))

---

**FUNCTION INDEX**

RECORD-ACCESS-MACRO .....3 RECORD-CONSTRUCTOR-MACRO 3 RECORD-PREDICATE-MACRO ..3 SETF-RECORD-ACCESS-MACRO 3

---

**MACRO INDEX**

ONCE-ONLY .....1 RECORD-CREATE .2 RECORD-FETCH ..2 RECORD-FFETCH .2 SETF-FETCH ....2 SETF-FFETCH ...2

---

**DEFINER INDEX**

DEFINE-FILE-ENVIRONMENT .....1 DEFINE-RECORD .....2

---

**SETF INDEX**

RECORD-FETCH .....2 RECORD-FFETCH .....2

---

**PROPERTY INDEX**

IL:XCL-EXTRAS .....4

---

**DEFINE-TYPE INDEX**

FILE-ENVIRONMENTS .....1

---