V10. fRec Record

The fRec record is a global record used by core FaceWare modules (FaceIt, DialIt, UtilIt, ViewIt). It is defined for program use in the include file "FaceStorXY". A byte offset, variable type (as integer or char), and brief description is provided below for each fRec element. WARNING: Do not assume that the content of fRec variables prefixed by "u", "w", or "c" is preserved across calls to the Control Manager or the FaceIt dispatching procedure. One exception: Most UtilIt commands preserve the "w" and "c" variables.

byte	type	name & description
0	int*2(8)	fHead FCMD record header
16	int*4	fOffPort off-screen GrafPort
20	int*4	fCOffPort off-screen CGrafPort*
24	int*4	fActiveWnd active WindowPtr**
28	int*4	fActiveRec active record address
32	int*2	fActiveID active FCMD baseID
34	int*2	fActiveResID active resID = FWND ID if ViewIt window
36	int*2	fMsgCount FaceIt message count
38	int*4	fFlags FaceIt bit flags
42	int*4	fStuff private FaceIt storage
46	int*4	fHeapBuff free memory buffer
50	int*4	fEnvFlags environment bit flags†
54	int*4	fSleep current WNE sleep value
58	int*2(4)	fScreenRect global screen bounds
66	int*2(4)	fDragRect global drag bounds
74	int*4	fFrontSleep WNE sleep at front
78	int*4	fBackSleep WNE sleep at back
82	int*2	fFiller1 reserved for future use
84	int*4	fStdBlock labeled item block
88	int*2	fStdCount labeled item count
90	int*2(8)	fSysEnvArr System Env. Record
106	int*2	fApplVRef working directory ref. number at time DoInit is called
108	char*4	fCreator application creator
112	int*4	fRefCon for program's use
116	int*2(7)	fFiller2 reserved for future use
130	int*2	fCursor current cursor ID
132	int*4	fFontMenu Font menu MenuHandle
136	int*4	fSizeMenu Size menu MenuHandle
140	int*4	fStyleMenu Style menu MenuHandle
144	int*4	fColorMenu Color menu MenuHandle
148	int*1	fFiller3 reserved for future use
149	int*1	fI1Err SToNum I1 error value
150	int*2	fI2Err SToNum I2 error value
152	int*4	fI4Err SToNum I4 error value
156	int*8	fI8Err SToNum I8 error value
164	real*4	fR4Err SToNum R4 error value
168	real*8	fR8Err SToNum R8 error value
176	real*10	fR10Err SToNum R10 error value
186	real*12	fR12Err SToNum R12 error value
198	int*2(125)	fFiller4 for future use
448	int*4	fWDEF custom WDEF code address
452	int*4	fGlueData for glue code's use††
456	int*4	fWaitNextEvent proc address
460	int*4	fNewWindow proc address
464	int*4	fNewCWindow proc address
468	int*4	fGetNewWindow proc address
472	int*4	fGetNewCWindow proc address
476	int*4	fGetNewDialog proc address
480	int*4	fDisposeWindow proc address

```
484   int*4      fDisposDialog proc address
488   int*4      fActiveWindow proc address
492   int*4      fSelectWindow proc address
496   int*4      fUpdateOther proc address
500   int*4(9) fFiller5 reserved for future use
536   int*2(8) fEvent FaceIt EventRecord
••• DialIt subrecord - obsolete •••
552   int*2(8) dHead
568   int*4      dlogptr
572   int*4      dataptr
576   int*4      dFlags
580   int*2      dBaseID
582   int*2      dPopID
584   int*2(20)   listID
624   real*4(20) slope
704   real*4(20) intercept
784   int*2(99)   dialog
982   int*2      dHit
984   int*2      dClick
986   int*2(8) dEvent
••• UtilIt subrecord •••
1002 int*2(8) uHead FCMD record header
1018 int*2      uI2 scratch 2-byte integer
1020 int*4      uI4 scratch 4-byte integer
1024 real*4     uR4 scratch 4-byte real
1028 real*8     uR8 scratch 8-byte real
1036 real*10   uR10 scratch 10-byte real
1046 real*12   uR12 scratch 12-byte real
1058 int*2(2) uPt scratch Point
1062 int*2(4) uRect scratch Rect
1070 int*4      uCommand FCMD command
1074 int*4(4) uParam FCMD command parameters
1090 int*4      uResult FCMD command result
1094 int*4      uMenuID scratch menuID number
1098 int*4      uMenuItem scratch menu item #
1102 char*256 uString scratch string
1358 char*256 uName scratch string
1614 int*2(3) uRGB scratch RGBColor
1620 int*1      uStyle scratch style
1621 int*1      uI1 scratch 1-byte integer
1622 int*8      uI8 scratch 8-byte integer
1630 int*4      uMenuHdl scratch MenuHandle
••• ViewIt subrecord •••
1634 int*2(8) vHead FCMD record header
1650 int*2      vErr error returned by ViewIt
1652 int*4      vCDEF custom CDEF code address
1656 int*4   vSelectCtl selected control handle
1660 int*4   vSelectRec selected record address
1664 int*2      vSelectID selected FCMD baseID
-- info returned by enabled items --
1666 int*2      wiHit item number
1668 int*2      wvHit view number
1670 int*2      wcHit control number (in view)
1672 int*2      wClick click type (1 = single,
                 2 = double, 3 = triple click)
1674 int*2(8) wEvent ViewIt EventRecord
-- window info returned by GetWnd --
1690 int*4      wWindow window's window pointer
1694 int*2      wResID associated FWND ID
1696 int*2      wCount total # of windows found
                 that match a & b in GetWnd call
```

1698 int*2 wvCount # of views in window
1700 int*2 wiCount # of controls in window
-- control info returned by GetCtl --
1702 int*4 cControl control's control handle
1706 int*2 ciIndex control's item number
1708 int*2 cvIndex parent view number
1710 int*2 ccIndex control number in view
1712 int*2 cBaseID driver baseID or CDEF ID
-- info copied from "cControl" block --
1714 int*4 cNext next control in list
1718 int*4 cOwner parent window pointer
1722 int*2(4) cRect control bounds
1730 int*1 cVis visible state (0 = hidden)
1731 int*1 cHilite hilite (-1 = inactive)
1732 int*2 cValue control value
1734 int*2 cMin minimum value
1736 int*2 cMax maximum value
1738 int*4 cInfo supplemental record handle
1742 int*4 cLoData for driver's use
1746 int*4 cAction action ProcPtr
1750 int*4 cRefCon for program's use
1754 char*256 cTitle control title
-- info copied from "cInfo" block --
2010 int*2(6) cStuff private ViewIt storage
2022 int*4 cTmplRefCon original RefCon
2026 int*4 cPtr shared record address
2030 int*4 cHiData for driver's use
2034 int*4 cView parent view control handle
2038 int*2(4) cOldRect private ViewIt storage
2046 int*2(4) cClip visible content area
2054 int*2(4) cContent content area
2062 int*2(4) cLimit control bounds limits
2070 int*4 cType control type (bit flags)
2074 int*2 cVarCode variation code
2076 char*4 cResType linked resource type
2080 int*2 cResID linked resource ID
2082 int*4 cResHdl linked resource handle
2086 int*4 cPrivate private ViewIt storage
2090 int*1 cFiller reserved for future use
2091 int*1 cCmdKey command key equivalent
2092 int*2 cPnRound rounded frame value
2094 int*2(2) cPnSize frame thickness
2098 int*1 cTxJust content justification
2099 int*1 cTxFace content text style
2100 int*2 cTxSize content text size
2102 int*2 cTxFont content text font
2104 int*2 cDataType linked data type
2106 int*1 cDataDigits digits to show
2107 int*1 cDataFormat format to use
2108 int*2 cDataOffset offset into record
2110 int*4 cDataPtr address of linked data
2114 int*2 cStorType control's data type
2116 int*4 cStorPtr scratch variable address
2120 int*4 cColors control's CCTabHandle
2124 int*4 cOverride override proc address
2128 int*2 cCount # of daughter controls
2130 char*256 cString string for driver's use
••• Local FCMD record & proc address table •••
2386 int*2 xEntries # of table entries
2388 int*4(40) xTable record & proc entries

2548 bytes total

* The four bytes in memory after the CGrafPort record contain a handle to the associated device, which will be the deepest device at launch time.

** The "Active" elements identify the active window, and the "Select" elements (in the ViewIt subrecord) identify the selected control within the active modeless or the top modal window. The "Active" elements are zeroed when either no window is active, or a modal ViewIt window is open. The "Select" elements are zeroed if no control is selected.

† Information about current environment. For example, "BitTst(@fEnvFlags,28)" checks for Color QuickDraw.
bit 0 (offset 31) = WaitNextEvent available
bit 1 (offset 30) = MultiFinder memory calls available
bit 2 (offset 29) = FPU (coprocessor) available
bit 3 (offset 28) = Color QuickDraw available
bit 4 (offset 27) = 32-bit QuickDraw available
bit 5 (offset 26) = System 7.0 or greater in use

†† "Glue code" refers to code that allows FaceWare modules to be used with high-level programming environments like HyperCard or ProGraph.