

# OBJECTIVE-C FOUNDATION CLASSES

## REFERENCE CARD

### Part 3: Wrappers

---

#### DatExit

##### Methods

+ add :(id) obj.....Add object to be freed during exit  
+ remove :(id) obj Remove object from being freed during exit

#### DCRC32

##### Methods

- init.....Init to start crc  
- init :(char \*) cstring.....Init and update with cstring  
- init :(uchar \*) data :(ulong) length  
    | Init and update with data  
- update :(char \*) cstring.....Calculate crc on cstring  
- update :(uchar \*) data :(ulong) length  
    | Calculate crc on data  
- (unsigned long) crc32.....Return the crc32 value

#### DDbm

##### Methods

- init.....Init without database  
- init :(char \*) name :(char \*) mode....Init with database  
- free.....Close database and free object  
- (BOOL) isOpen.....Check for open database  
- (BOOL) isReadOnly.....Check for read only database  
- (int) error.....Return last error  
- (BOOL) open :(char \*) name :(char \*) mode Open database  
- (BOOL) insert :(void\*) key :(uint) klen :(void\*) data  
    |:(uint) dlen....Insert (with replace) the data for the key  
- (NSData \*) get :(void\*) key :(uint) klen  
    | Fetch data for key  
- (BOOL) has :(void \*) key :(uint) klen....Check for key  
- (BOOL) delete :(void \*) key :(uint) klen....Delete key  
- (BOOL) reorganize.....Reorganize the database  
- (DDbm \*) close.....Close the database  
- (DList \*) keys.....Return list with all keys in database  
- (DList \*) objects.....Return list with all data in database

#### DDirectory

##### Classmethods

+ (BOOL) isDriveSeparator :(char) ch  
    | Check for drive separator  
+ (BOOL) isPathSeparator :(char) ch  
    | Check for path separator  
+ (BOOL) create :(char \*) path.....Create directory  
+ (BOOL) move :(char \*) path :(char \*) newPath  
    | Move/Rename directory  
+ (BOOL) delete :(char \*) path.....Delete directory  
+ (DDirectory \*) current.....Return current working dir  
+ (BOOL) current :(char \*) path.....Set current dir  
+ (DDirectory \*) temp.....Return temp directory  
+ (BOOL) exist :(char \*) path.....Check if dir exists  
+ (DList \*) childS :(char \*) path :(id) filter  
    | Return names in directory  
+ (int) error.....Return last error  
*Objectmethods*  
- init.....Init empty directory  
- init :(char \*) path.....Init with directory  
- deepen.....Deepen copied object  
- (BOOL) isAbsolute.....Check for absolute path  
- (BOOL) isEmpty.....Check for empty path  
- (char) drive.....Return drive letter or EOS  
- path :(char \*) path.....Set directory path  
- (char \*) path.....Get directory path  
- (char \*) name.....Get last directory name  
- child :(char \*) name.....Add subdirectory  
- (BOOL) parent.....Move to parent directory  
- (DList \*) names.....Split path in list of names  
- names :(DList \*) names.....Build path from names

#### DBZipFile

##### Methods

- init.....Init to empty file object  
- init :(char \*) name :(char \*) mode.....Open bz2 file  
- init :(char \*) name :(char \*) mode :(int) small  
    | Open bz2 file with memory usage indication  
- free.....Free the object (close the file)  
- (int) error.....Return the last error  
- (BOOL) isOpen.....Check for open file  
- (BOOL) open :(char \*) name :(char \*) mode..Open bz2 file  
- open :(char \*) name :(char \*) mode :(int) small  
    | Open bz2 file with memory usage indication  
- (BOOL) isEof.....Check for end-of-file  
- (char) readChar.....Read a character

- (DText \*) readLine.....Read a line  
- (DText \*) readText.....Read all text  
- (DText \*) readText :(long) len.....Read len text  
- (BOOL) writeChar :(char) ch.....Write character  
- (BOOL) writeLine :(char \*) text.....Write line  
- (BOOL) writeText :(char \*) text.....Write text  
- (uchar) readByte.....Read a byte  
- (NSData \*) readData :(ulong) length  
    | Read a data string  
- (double) readDouble.....Read a double  
- (long) readLong.....Read a long  
- (short) readShort.....Read a short  
- (BOOL) writeByte :(uchar) byte.....Write a byte  
- (BOOL) writeData :(uchar \*) text :(ulong) length  
    | Write a data string  
- (BOOL) writeDouble :(double) nr.....Write a double  
- (BOOL) writeLong :(long) nr.....Write a long  
- (BOOL) writeShort :(short) nr.....Write a short  
- (DList \*) readLines.....Read all lines in a list  
- (BOOL) writeLines :(DList \*) list.....Write list to file  
- close.....Close the file

#### DGZipFile

##### Constants

DGZ\_SEEK\_SET.....Seek from start of file  
DGZ\_SEEK\_CUR.....Seek from current position  
DGZ\_DEFAULT.....Default strategy  
DGZ\_FILTERED.....Filtered data strategy  
DGZ\_HUFFMAN.....Huffman compression strategy

##### Methods

- init.....Init to empty file object  
- init :(char \*) name :(char \*) mode.....Open file  
- init :(char \*) name :(char \*) mode :(int) level  
    |:(int) strategy.....Open file with level and strategy  
- free.....Free the object (close the file)  
- (int) error.....Return the last error  
- (BOOL) isOpen.....Check for open file  
- (BOOL) open :(char \*) name :(char \*) mode....Open file  
- open :(char \*) name :(char \*) mode :(int) level  
    |:(int) strategy.....Open file with level and strategy  
- (BOOL) isEof.....Check for end-of-file  
- (char) readChar.....Read a character  
- (DText \*) readLine.....Read a line  
- (DText \*) readText.....Read all text

```
- (DText *) readText :(long) len.....Read len text
- (BOOL) seek :(ulong) off :(int) org.....Move position
- (BOOL) skip :(ulong) off .....Skip forward
- (unsigned long) tell .....Return current position
- (BOOL) writeChar :(char) ch .....Write character
- (BOOL) writeLine :(char *) text .....Write line
- (BOOL) writeText :(char *) text .....Write text
- (uchar) readByte .....Read a byte
- (DData *) readData :(ulong) length
    | Read a data string
- (double) readDouble .....Read a double
- (long) readLong .....Read a long
- (short) readShort .....Read a short
- (BOOL) writeByte :(uchar) byte .....Write a byte
- (BOOL) writeData :(uchar *) text :(ulong) length
    | Write a data string
- (BOOL) writeDouble :(double) nr .....Write a double
- (BOOL) writeLong :(long) nr .....Write a long
- (BOOL) writeShort :(short) nr .....Write a short
- (DList *) readLines .....Read all lines in a list
- (BOOL) writeLines :(DList *) list .....Write list to file
- (BOOL) flush .....Flush the output buffers
- close .....Close the file
```

DInet6SocketAddress

*Constants*

DSA\_AF\_INET6 ..... Inet6 socket family (IPv6)

*Methods*

```
- init .....Init empty inet socket address
- init :(ulong) a1 :(ulong) a2 :(ulong) a3 :(ulong) a4
    |:(int) port :(ulong) flowInfo :(ulong) scopeId
    | Init with IPv6 address
- init ::(uchar[16]) address :(int) port :(ulong)
    | flowInfo :(ulong) scopeId .....Init with IPv6 address
- (int) error .....Get the last error
- (int) family .....Return the family
- (void *) sockaddr .....Return the sockaddr struct
- (int) size .....Return the size of sockaddr
- (int) port .....Return the port of the address
- (DText *) host .....Return the host name
- set :(ulong) a1 :(ulong) a2 :(ulong) a3 :(ulong) a4
    |:(int) port :(ulong) flowInfo :(ulong) scopeId
    | Set with IPv6 address
- init ::(uchar[16]) address :(int) port :(ulong)
```

```
    | flowInfo :(ulong) scopeId .....Set with IPv6 address
    |:(int) port .....Set with b1.b2.b3.b4
- (BOOL) host :(char *) name :(int) port :(ulong)
    | flowInfo :(ulong) scopeId .....Set with host
- (BOOL) sockaddr :(void *) addr :(int) size
    | Set with sockaddr struct
- loopback :(int) port :(ulong) flowInfo :(ulong)
    | scopeId .....Set with loopback
- any :(int) port :(ulong) flowInfo :(ulong) scopeId
    | Set with any address
- close .....Close the address
```

DInetSocketAddress

*Constants*

DSA\_AF\_INET ..... Inet socket family

*Methods*

```
- init .....Init empty inet socket address
- init :(long) address :(int) port .....Init with address
- init :(uchar) b1 :(uchar) b2 :(uchar) b3 :(uchar) b4
    |:(int) port .....Init with b1.b2.b3.b4
- (int) error .....Get the last error
- (int) family .....Return the family
- (void *) sockaddr .....Return the sockaddr struct
- (int) size .....Return the size of sockaddr
- (int) port .....Return the port of the address
- (DText *) host .....Return the host name
- set :(ulong) address :(int) port .....Set with address
- set :(uchar) b1 :(uchar) b2 :(uchar) b3 :(uchar) b4
    |:(int) port .....Set with b1.b2.b3.b4
- (BOOL) host :(char *) name :(int) port ....Set with host
- (BOOL) sockaddr :(void *) addr :(int) size
    | Set with sockaddr struct
- loopback :(int) port .....Set with loopback
- any :(int) port .....Set with any address
- broadcast :(int) port .....Set with broadcast address
- close .....Close the address
```

DKey

*Constants*

DKEY\_NULL ..... ctrl-@ key

DKEY\_BACKSPACE ..... Backspace key

DKEY\_ENTER ..... Enter key

DKEY\_ESCAPE ..... Escape key

DKEY\_DELETE ..... Delete key

DKEY\_F1 ..... Function key 1

DKEY\_F2 ..... Function key 2

DKEY\_F3 ..... Function key 3

DKEY\_F4 ..... Function key 4

DKEY\_F5 ..... Function key 5

DKEY\_F6 ..... Function key 6

DKEY\_F7 ..... Function key 7

DKEY\_F8 ..... Function key 8

DKEY\_F9 ..... Function key 9

DKEY\_F10 ..... Function key 10

DKEY\_F11 ..... Function key 11

DKEY\_F12 ..... Function key 12

DKEY\_F13 ..... Function key 13

DKEY\_F14 ..... Function key 14

DKEY\_F15 ..... Function key 15

DKEY\_F16 ..... Function key 16

DKEY\_F17 ..... Function key 17

DKEY\_F18 ..... Function key 18

DKEY\_F19 ..... Function key 19

DKEY\_F20 ..... Function key 20

DKEY\_NUMLOCK ..... Numlock key

DKEY\_CAPSLOCK ..... Capslock key

DKEY\_SCROLLLOCK ..... Scroll key

DKEY\_SHIFT ..... Shift key

DKEY\_CTRL ..... Control key

DKEY\_ALT ..... Alt key

DKEY\_UP ..... Up arrow key

DKEY\_DOWN ..... Down arrow key

DKEY\_RIGHT ..... Right arrow key

DKEY\_LEFT ..... Left arrow key

DKEY\_INSERT ..... Insert key

DKEY\_HOME ..... Home key

DKEY\_END ..... End key

DKEY\_PAGEUP ..... Page up key

DKEY\_PAGEDOWN ..... Page down key

DKEY\_MOUSE\_KEYS ..... Mouse keys group bit

DKEY\_MOUSE\_RIGHT ..... Right mouse button

DKEY\_MOUSE\_MIDDLE ..... Middle mouse button

DKEY\_MOUSE\_LEFT ..... Left mouse button

DKEY\_MOD\_SHIFT ..... shift modifier

DKEY\_MOD\_CTRL ..... control modifier

DKEY\_MOD\_ALT ..... alt modifier

DKEY\_MOD\_KP ..... keypad modifier

*Methods*

- `init` ..... Init to NULL key
- `init :(int) code` ..... Init to key code
- `copy` ..... Copy the object
- `free` ..... Free the object
- `(BOOL) isCtrlKey` ..... Check for ctrl key
- `(BOOL) isFunctionKey` ..... Check for function key
- `(BOOL) isKeypadKey` ..... Check for keypad key
- `(BOOL) isAltKey` ..... Check for alt key
- `(BOOL) isShiftKey` ..... Check for shift key
- `(BOOL) isMouseKey` ..... Check for mouse key
- `(DKey *) set :(int) code` ..... Set the code for the key
- `(int) get` ..... Get the key code
- `(int) key` ..... Get the key (without modifiers)
- `(int) mod` ..... Get the key modifiers
- `(DText *) toText` ..... Convert to text object
- `(int) fromString :(char **) cstr` ..... Parse from string

DMD5

*Methods*

- `init` ..... Init empty md5
- `init :(char *) cstr` ..... Init with c-string
- `init :(uchar *) data :(ulong) len` ..... Init with data
- `update :(char *) cstr` ..... Update with c-string
- `update :(uchar *) data :(ulong) len` ..... Update with data
- `(DData *) digest` ..... Return digest
- `(DText *) hexdigest` ..... Return digest in hex-ascii

DRegex

*Syntax*

- ..... Match any char (incl. newline)
- `*` ..... Match zero or more
- `+` ..... Match one or more
- `?` ..... Match zero or one
- `{c}` ..... Match exactly c times
- `{min,max}` ..... Match min..max times
- `|` ..... Match alternatives
- `[]` ..... Match one in the list
- `[^]` ..... Match any except in list
- `[::]` ..... Match a class in a list
- `()` ..... Group or subexpression
- `^` ..... Match begin of line
- `$` ..... Match end of line

*Constants*

- `DRE_NO_MATCH` ..... No result for match or search
- `DRE_ERROR` ..... Error for match or search

*Methods*

- `init` ..... Init empty regex
- `init :(char *) pattern` ..... Init with case sensitive pattern
- `free` ..... Free the regex
- `(BOOL) ccompile :(char *) ptrn` ..... Compile case sens. pattern
- `(BOOL) icodepile :(char *) ptrn`
  - | Compile case insens. pattern
- `(int) match :(char *) str` ..... Match for length
- `(int) match :(char *) str :(int) from`
  - | Match for length with offset
- `(int) match :(uchar *) data :(int) length :(int) from`
  - | Match for length with offset
- `(int) search :(char *) str` ..... Search for start
- `(int) search :(char *) str :(int) from :(int) to`
  - | Search for start in range
- `(int) search :(uchar *) data :(int) len` ..... Search for start
- `(int) search :(uchar *)str :(int)len :(int)fr :(int)to`
  - | Search for start in range
- `(DArray *) indices` ..... Last matched indices
- `(DArray *) matches :(char *) str` ..... Last matched texts
- `(DArray *) matches :(uchar *) data :(int) len`
  - | Last matched data

DSHA1

*Methods*

- `init` ..... Init empty sha1
- `init :(char *) cstr` ..... Init with c-string
- `init :(uchar *) data :(ulong) len` ..... Init with data
- `update :(char *) cstr` ..... Update with c-string
- `update :(uchar *) data :(ulong) len` ..... Update with data
- `(DData *) digest` ..... Return digest
- `(DText *) hexdigest` ..... Return digest in hex-ascii

DSocket

*Constants*

- `DSK_STREAM` ..... Stream socket type
- `DSK_DGRAM` ..... Datagram socket type
- `DSK_MSG_OOB` ..... Out of band message flag
- `DSK_MSG_DONTROUTE` ..... Do not route the message flag
- `DSK_MSG_PEEK` ..... Peek data message flag
- `DSK_MSG_WAITALL` ..... Wait for the full message flag

- `DSK_SHUT_RD` ..... Shutdown receive flag
- `DSK_SHUT_WR` ..... Shutdown send flag
- `DSK_SHUT_RDWR` ..... Shutdown both send and receive flag

*Classmethods*

- + `(int) protocol :(const char *) name`

*Methods*

- `init` ..... Init empty socket
- `init :(int) fileno :(id) address :(int) type`
  - |:(int) protocol ..... Init socket with file descr.
- `init :(int) family :(int) type :(int) protocol`
  - | Init socket with address
- `free` ..... Free socket
- `(int) fileno` ..... Return file descriptor
- `(int) error` ..... Return last error
- `(BOOL) setSocketOption :(int) level :(int) optname`
  - |:(void \*) optval :(int) optsize ..... Set socket option
- `(BOOL) getSocketOption :(int) level :(int) optname`
  - |:(void \*) optval :(int) optsize ..... Get socket option
- `(BOOL) blocking :(BOOL) block` ..... Set blocking state
- `(BOOL) blocking` ..... Return blocking state
- `(BOOL) reuseAddr :(BOOL) reuse` ..... Set reuse state
- `(BOOL) reuseAddr` ..... Get reuse state
- `(BOOL) sendBufferSize :(int) size` ..... Set send buffer size
- `(int) sendBufferSize` ..... Get send buffer size
- `(BOOL) receiveBufferSize :(int) size`
  - | Set receive buffer size
- `(int) receiveBufferSize` ..... Get receive buffer size
- `(BOOL) keepAlive :(BOOL) keep` ..... Set keep alive state
- `(BOOL) keepAlive` ..... Get keep alive state
- `(BOOL) linger :(unsigned) secs` ..... Set linger time
- `(unsigned) linger` ..... Get linger time
- `(BOOL) open :(int) family :(int) type :(int) protocol`
  - | Open socket with address
- `(BOOL) bind :(id) address` ..... Bind socket to address
- `(BOOL) listen :(int) backlog` ..... Listen for connections
- `(DSocket *) accept` ..... Accept a connection
- `(BOOL) connect :(id) address` ..... Connect to address
- `(BOOL) close` ..... Close socket
- `(BOOL) shutdown :(int) what` ..... Shutdown connection
- `(int) sendto :(id) address :(void *) data`
  - |:(int) length :(int) flags ..... Send data connectionless
- `(DData *) recvfrom :(id) address :(int) length`
  - |:(int) flags ..... Receive data connectionless
- `(int) send :(void *) data :(int) length :(int) flags`
  - | Send data via connection

- (DData \*) recv :(int) length :(int) flags  
  - | Receive data via connection
- (int) sendto :(id) address :(char \*) cstr :(int) flags  
  - | Send text connectionless
- (int) recvfrom :(Text \*) dest :(id) address :(int) len  
  - |:(int) flags.....Receive text connectionless
- (int) send :(char \*) cstring :(int) flags  
  - | Send text via connection
- (int) recv :(DText \*) dest :(int) length :(int) flags  
  - | Receive text via connection

## DSystemLogger

- init.....Init default system logger
- init :(char \*) appName :(BOOL) toStdErr  
  - | Init system logger with application name
- (BOOL) doLog :(int) level :(char \*) message  
  - | Log a message with a level
- (int) mask :(int) levels.....Set the level mask
- (int) mask :(int) high :(int) low.....Set the level range

Note: Only where available (unix)

## DTextDrawable

- init.....Init text drawable
- free.....Free text drawable
- (unsigned) cursorX.....Get cursor x-position
- (unsigned) cursorY.....Get cursor y-position
- (BOOL) cursor :(unsigned) x :(unsigned) y  
  - | Set cursor position
- (int) cursor :(int) state.....Set cursor state
- (int) pointer :(int) state.....Set pointer state
- (int) error.....Get the last error
- (unsigned long) tell.....Tell current cursor position
- (BOOL) seek :(unsigned long) offset :(int) origin  
  - | Set the cursor position
- (BOOL) skip :(unsigned long) offset..Skip cursor positions
- (unsigned) maxX.....Get maximum x-position
- (unsigned) lines.....Get number of lines
- (unsigned) maxY.....Get maximum y-position
- (unsigned) columns.....Get number of columns
- drawable :(unsigned) columns :(unsigned) lines  
  - | Set the size of the drawable
- (BOOL) isValid :(unsigned) x :(unsigned) y  
  - | Check if position is on the drawable

- (BOOL) clip :(unsigned) minX :(unsigned) maxX  
  - |:(unsigned) minY :(unsigned) maxY.....Set clipping area
- (BOOL) clip.....Set clipping area full drawable
- (unsigned) clipMinY.....Get minimum clipping y-position
- (unsigned) clipMaxY.....Get maximum clipping y-position
- (unsigned) clipMinX.....Get minimum clipping x-position
- (unsigned) clipMaxX.....Get maximum clipping x-position
- (BOOL) color :(DColor \*) fgcolor :(DColor \*) bgcolor  
  - | Set foreground and background color
- (BOOL) foregroundcolor :(DColor \*) fgcolor  
  - | Set foreground color
- (DColor \*) foregroundcolor.....Get foreground color
- (BOOL) backgroundcolor :(DColor \*) bgcolor  
  - | Set background color
- (DColor \*) backgroundcolor.....Get background color
- (BOOL) startDrawing.....Start drawing on text screen
- (BOOL) startDrawing :(unsigned) minX :(unsigned) maxX  
  - |:(unsigned) minY :(unsigned) maxY  
    - | Start drawing in clipped area
- stopDrawing.....Stop drawing
- (BOOL) isDrawing.....Check for drawing mode
- (BOOL) clear.....Clear the drawable
- (BOOL) writeText :(char \*) text...Write text on drawable
- (BOOL) writeChar :(char) ch..Write character on drawable
- (BOOL) writeLine :(char \*) textWrite text line on drawable
- (BOOL) writeText :(unsigned) startX :(unsigned) startY  
  - |:(const char \*) text.....Write text on drawable
- (BOOL) writeChar :(unsigned) startX :(unsigned) startY  
  - |:(char) ch.....Write character on drawable
- (int) readChar :(unsigned) startX :(unsigned) startY  
  - | Read a character from the drawable
- (BOOL) drawHLine :(unsigned) startX : (unsigned) endX  
  - |:(unsigned) startY :(int) lineType  
    - |:(unsigned) lineWidth..Draw horizontal line on drawable
- (BOOL) drawHLine :(unsigned) startX : (unsigned) endX  
  - |:(unsigned) startY.....Draw horizontal line on drawable
- (BOOL) drawHLine :(unsigned) endX...Draw horizontal line
- (BOOL) drawVLine :(unsigned) startX :(unsigned) startY  
  - |:(unsigned) endY :(int) lineType  
    - |:(unsigned) lineWidth....Draw vertical line on drawable
- (BOOL) drawVLine :(unsigned) startX :(unsigned) startY  
  - |:(unsigned) endY.....Draw vertical line on drawable
- (BOOL) drawVLine :(unsigned) endY.....Draw vertical line
- (BOOL) drawLine :(unsigned) startX :(unsigned) endX  
  - |:(unsigned) startY :(unsigned) endY :(int) lineType

- |:(unsigned) lineWidth....Draw vertical line on drawable
- (BOOL) drawLine :(unsigned) startX :(unsigned) endX  
  - |:(unsigned) startY :(unsigned) endY...Draw vertical line
- (BOOL) drawLine :(unsigned) endX :(unsigned) endY  
  - | Draw vertical line on drawable
- (BOOL) drawPoint :(unsigned) startX :(unsigned) startY  
  - | Draw point on drawable
- (BOOL) drawPoint.....Draw point on drawable
- (BOOL) blit :(unsigned) startX :(unsigned) startY  
  - |:(DTextDrawable \*) other :(unsigned) oStartX  
    - |:(unsigned) oEndX :(unsigned) oStartY  
      - |:(unsigned) oEndY...Blit other drawable on this drawable

## DTextScreen : DTextDrawable

- init.....Init text screen object
- init :(DColor \*) fgcolor :(DColor \*) bgcolor  
  - | Init and open text screen
- free.....Free the text screen
- (BOOL) isOpen.....Check if text screen is open
- (BOOL) hasColors.....Check if text screen support colors
- (id) screenHeight :(id) handler..Set screen event handler
- (id) screenHeight.....Get current screen event handler
- (BOOL) open :(DColor \*) fgcolor :(DColor \*) bgcolor  
  - | Open text screen
- close.....Close text screen
- beep.....Generate beep
- (int) cursor :(int) state.....Set the state of the cursor
- (BOOL) startDrawing.....Start drawing on the text screen
- stopDrawing.....Stop drawing on the text screen
- (unsigned) waitEvents.....Wait and process events
- (unsigned) processEvents.....Processing waiting events
- clearEvents.....Clear any pending events

## DTextSurface : DTextDrawable

- init.....Init text surface
- free.....Free text surface
- (BOOL) isOpen.....Check if surface is open
- (BOOL) open :(unsigned) columns :(unsigned) lines  
  - |:(DColor \*) fgcolor :(DColor \*) bgcolor..Open the text surface
- close.....Close the text surface

## DTimer

*Methods*

```
+ (void) delay :(long) msec.....Delay msec
- init.....Init default timer
- init :(long) timeout.....Init with time-out value
- (long) timer.....Return current timer
- (long) timeout.....Return current time out value
- timeout :(long) timeout.....Set time out value
- restart.....Restart the timer
- (BOOL) isExpired.....Test for expired timer, auto restart
- (BOOL) isExpired :(long) timeout
    | Test for timed expiration, auto restart
```

Note: All times in milliseconds

## DUnixSocketAddress

### Constants

```
DSA_AF_UNIX.....Unix socket family
```

### Methods

```
- init.....Init empty unix socket address
- init :(char *) filename.....Init with filename
- (int) error.....Get the last error (always 0)
- (int) family.....Return the family
- (void *) sockaddr.....Return the sockaddr struct
- (int) size.....Return the size of sockaddr
- (int) port.....Return the port
- (DText *) host.....Return the host name
- (BOOL) filename :(char *) name.....Set with filename
- (BOOL) sockaddr :(void *) addr :(int) size
    | Set with sockaddr struct
- close.....Close the address
```

Note: Only where available (unix)

## DXMLReader

```
- init.....Init xml reader
- deepen.....Deepen copied object
- free.....Free xml reader
- bufferSize :(int) size.....Set parser buffer size
- (int) bufferSize.....Get parser buffer size
- encoding :(char *) encoding.....Set override encoding
- (char *) encoding.....Get override encoding
- (BOOL) parse :(id) source :(char *) name :(id) handler
    |:(char) separator.....Parse the xml source with handler
- (int) lineNumber.....Return the parsed line number
- (int) columnNumber.....Return the parsed column number
- (const char *) name.....Return the name of the source
```

```
+ (char *) errorToString :(int) error.....Translate error
```