

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` 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` Set socket option
- `(BOOL) getSocketOption :(int) level :(int) optname` 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` Send data connectionless
- `(DData *) recvfrom :(id) address :(int) length` Receive data connectionless
- `(int) send :(void *) data :(int) length :(int) flags` Send data via connection


```
+ (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
```