

# OBJECTIVE-C FOUNDATION CLASSES

## REFERENCE CARD

### Part 4: Network

#### DHTTPClient

##### Constants

DHTTP\_PORT ..... Default HTTP port  
DHC\_HEAD ..... HTTP Head request  
DHC\_GET ..... HTTP Get request

##### Methods

- **init** ..... Initialise a HTTP client (def. version: 1.1)  
- **init :(int) major :(int) minor ....** Init client with version  
- **free** ..... Free the HTTP client  
- **(DSocket \*) socket** ..... Return the socket of the client  
- **(BOOL) sendStartRequest :(int) request :(DURL \*) path**  
    | Send a HTTP request to the server in the URL  
- **(BOOL) sendStartRequest :(int) request :(DURL \*) path**  
    | **:(char \*) proxy :(int) proxyPort** Send request via proxy  
- **(BOOL) sendHeader :(char \*) header :(char \*) argument**  
    | Send a header after the HTTP request  
- **(BOOL) sendAcceptLanguageHeader :(char \*) languages**  
    | Send accept-language header  
- **(BOOL) sendUserAgentHeader :(char \*) agent**  
    | Send user accept header  
- **(BOOL) sendRefererHeader :(char \*) referer**  
    | Send referer header  
- **(BOOL) sendConnectionHeader :(char \*) connection**  
    | Send the nextconnection state: close or keep-alive  
- **(BOOL) sendEndRequest** ..... Send the end of the request  
- **(BOOL) receiveReply** ..... Receive a reply after a request  
- **(int) reason** ..... Get the reason in the reply  
- **(char \*) reasonText** ..... Get the reason text in the reply  
- **(int) peerMajor** ..... Return the major version of peer  
- **(int) peerMinor** ..... Return the minor version of peer  
- **(DText \*) header :(char \*) header**  
    | Get the argument of header in the reply  
- **(DHashIterator \*) headers** ..... Get all headers in the reply  
- **(const unsigned char \*) body** ..... Get the data in the body  
- **(unsigned long) bodyLength** ..... Get the length of the body

#### DTCPServer

##### Constants

DTS\_BLOCKING ..... Connections are blocking  
DTS\_THREADING ..... Connections in a thread

DTS\_FORKING ..... Connections in a child process  
*Methods*

- **init** ..... Init default tcp server  
- **init :(int) family :(int) prot ...** Init and open tcp server  
- **free** ..... Free tcp server  
- **(DSocket \*) socket** ..... Return socket of server  
- **sendFlag :(int) flag** ..... Set the send flag  
- **(int) sendFlag** ..... Return the send flag  
- **recvFlag :(int) flag** ..... Set the receive flag  
- **(int) recvFlag** ..... Return the receive flag  
- **recvLength :(unsigned) length** ..... Set the receive length  
- **(unsigned) recvLength** ..... Return the receive length  
- **(BOOL) open :(int) family :(int) prot ...** Open tcp server  
- **(BOOL) start :(int) addr :(int) backlog :(int) mode**  
    | **:(int) connections** ..... Start server, wait for connections  
- **handleConnection :(DSocket \*) peer** Handle new connection  
- **(BOOL) handleRequest :(DData \*) request**  
    | **:(DData \*) response** ..... Handle request from client

#### DTCPClient

##### Methods

- **init** ..... Init default tcp client  
- **init :(int) family :(int) protocol** ..... Init tcp client  
- **free** ..... Free tcp client  
- **(DSocket \*) socket** ..... Return socket of client  
- **(BOOL) isConnected** ..... Test for open connection  
- **sendFlag :(int) flag** ..... Set send flag  
- **(int) sendFlag** ..... Return send flag  
- **recvFlag :(int) flag** ..... Set the receive flag  
- **(int) recvFlag** ..... Return the receive flag  
- **(BOOL) open :(int) family :(int) prot ...** Open tcp client  
- **(BOOL) start :(id) address ...** Start connection with server  
- **(DData \*) doRequest :(uchar \*) request :(unsigned) len**  
    | **:(unsigned) respLen** ..... Send request and wait for response  
- **stop** ..... Stop connection

#### DTelNetClient

##### Constants

DTNC\_PORT ..... Default Telnet port  
DTNC\_IAC ..... Command: Interpret as command  
DTNC\_DONT ..... Command: Do not use this option  
DTNC\_DO ..... Command: Do use this option  
DTNC\_WONT ..... Command: Will not use this option

DTNC\_WILL ..... Command: Will use this option  
DTNC\_SB ..... Command: Start subnegotiation  
DTNC\_GA ..... Command: Go ahead  
DTNC\_EL ..... Command: Erase line  
DTNC\_EC ..... Command: Erase character  
DTNC\_AYT ..... Command: Are you there  
DTNC\_AO ..... Command: Abort output  
DTNC\_IP ..... Command: Interrupt process  
DTNC\_BRK ..... Command: Break  
DTNC\_DM ..... Command: Data mark  
DTNC\_NOP ..... Command: No operation  
DTNC\_SE ..... Command: End subnegotiation  
DTNC\_BINARY ..... Option: 8-bit data path  
DTNC\_ECHO ..... Option: Echo  
DTNC\_RCP ..... Option: Prepare to reconnect  
DTNC\_SGA ..... Option: Suppress go ahead  
DTNC\_NAMS ..... Option: Approximate message size  
DTNC\_STATUS ..... Option: Give status  
DTNC\_TM ..... Option: Timing mark  
DTNC\_RCTE ..... Option: Remote controlled transmission and echo  
DTNC\_NAOL ..... Option: Output line width  
DTNC\_NAOP ..... Option: Output page size  
DTNC\_NAOCRD ..... Option: CR disposition  
DTNC\_NAOHTS ..... Option: Horizontal tabstops  
DTNC\_NAOHTD ..... Option: Horizontal tab disposition  
DTNC\_NAOFFD ..... Option: Formfeed disposition  
DTNC\_NAOVTS ..... Option: Vertical tab stops  
DTNC\_NAOVTD ..... Option: Vertical tab disposition  
DTNC\_NAOLFD ..... Option: LF disposition  
DTNC\_XASCII ..... Option: Extended ascii character set  
DTNC\_LOGOUT ..... Option: Force logout  
DTNC\_BM ..... Option: Byte macro  
DTNC\_DET ..... Option: Data entry terminal  
DTNC\_SUPDUP ..... Option: Supdup protocol  
DTNC\_SUPDUPOUTPUT ..... Option: Supdup output  
DTNC\_SNDLOC ..... Option: Send location  
DTNC\_TTYPE ..... Option: Terminal type  
DTNC\_EOR ..... Option: End or record  
DTNC\_TUID ..... Option: TACACS user identification  
DTNC\_OUTMRK ..... Option: Output marking  
DTNC\_TTYLOC ..... Option: Terminal location number  
DTNC\_3270REGIME ..... Option: 3270 regime  
DTNC\_X3PAD ..... Option: X.3 PAD  
DTNC\_NAWS ..... Option: Window size  
DTNC\_TSPEED ..... Option: Terminal speed

DTNC\_LFLOW.....Option: Remote flow control  
DTNC\_LINEMODE ..... Option: Linemode option  
DTNC\_XDISPLOC ..... Option: X Display Location  
DTNC\_OLD\_ENVIRON ..... Option: Old - Environment variables  
DTNC\_AUTHENTICATION.....Option: Authenticate  
DTNC\_ENCRYPT ..... Option: Encryption option  
DTNC\_NEW\_ENVIRON ..... Option: Environment variables  
DTNC\_SERVER ..... Who: server  
DTNC\_CLIENT.....Who: client

#### *ClassMethods*

+ (char \*) commandToString :(uchar) command  
| Convert command to a string  
+ (char \*) optionToString :(int) option  
| Convert an option to a string

#### *ObjectMethods*

- init.....Init a telnet client  
- free.....Free the telnet client  
- (DSocket \*) socket ..... Return the socket of the client  
- (BOOL) isConnected ..... Check for connection with server  
- (BOOL) option :(int) who :(int) option  
| Check state of option  
- (BOOL) open :(id) server ..... Open connection to server  
- (BOOL) close.....Close connection to server  
- (BOOL) requestOpenNegotiation :(int) who :(int) option  
|:(BOOL) state . Put open negotiation request in send buffer  
- (BOOL) requestSubNegotiation :(int) option  
| Put sub negotiation request in send buffer  
- (BOOL) respondSubNegotiation :(int) option  
|:(char \*) data :(int) length  
| Put sub negotiation response in send buffer  
- (BOOL) respondNegotiation :(int) who :(int) option  
|:(BOOL) accepted .. Put negotiation response in send buffer  
- (BOOL) AYT ..... Put Are You There in send buffer  
- (BOOL) sendText :(const unsigned char \*) text  
| Put normal text in send buffer en send buffer to server  
- (BOOL) pendingRequests  
| Check for pending messages in send buffer  
- (BOOL) pendingNegotiations . Check for pending negotiations  
- (DData \*) receive ..... Receive data from server  
- (BOOL) receive :(DData \*) response  
| Receive data from server  
- (BOOL) processSpecialCommand :(unsigned char) command  
| Process commands received from server (to be overridden)  
- (BOOL) processOpenNegotiation :(int) who :(int) option  
|:(int) state .. Process received open negotiation (override)

- (BOOL) processResponseNegotiation :(int) who  
|:(BOOL) accepted :(int) option  
| Process received negotiation response (to be overridden)  
- (BOOL) processRequestSubNegotiation :(int) option  
| Process received sub negotiation request (to be overridden)  
- (BOOL) processResponseSubNegotiation :(int) option  
|:(char \*) data :(int) length  
| Process received sub negotiation response (to be overridden)

## DUDPServer

#### *Methods*

- init.....Init default udp server  
- init :(int) family :(int) prot...Init and open udp server  
- free ..... Free udp server  
- (DSocket \*) socket ..... Return socket of server  
- sendFlag :(int) flag.....Set the send flag  
- (int) sendFlag ..... Return the send flag  
- recvFlag :(int) flag.....Set the receive flag  
- (int) recvFlag.....Return the receive flag  
- recvLength :(unsigned) length ..... Set the receive length  
- (unsigned) recvLength.....Return the receive length  
- (BOOL) open :(int) family :(int) prot...Open udp server  
- (BOOL) start :(id) addr ..... Start server, blocking  
- (BOOL) handleRequest :(DData \*) request  
|:(DData \*) response.....Handle request from client

## DUDPClient

#### *Methods*

- init.....Init default udp client  
- init :(int) family :(int) protocol ..... Init udp client  
- free.....Free udp client  
- (DSocket \*) socket.....Return socket of client  
- sendFlag :(int) flag.....Set send flag  
- (int) sendFlag.....Return send flag  
- recvFlag :(int) flag.....Set the receive flag  
- (int) recvFlag.....Return the receive flag  
- (BOOL) open :(int) family :(int) prot...Open udp client  
- (BOOL) start :(id) address ..... Start connection  
- (DData \*) doRequest :(id) server :(uchar \*) request  
|:(unsigned) len :(unsigned) respLen  
| Send request and wait for response  
- stop.....Stop connection

## DURL

#### *Methods*

- init.....Init empty url  
- init :(char \*) url ..... Init with url  
- free.....Free url  
- deepen.....Deepen a copied url  
- (char \*) scheme ..... Return the scheme in the url  
- (char \*) protocol.....Return the protocol in the url  
- scheme :(char \*) scheme ..... Set the scheme in the url  
- (char \*) user.....Return the user in the url  
- user :(char \*) user ..... Set the user in the url  
- (char \*) password.....Return the password in the url  
- password :(char \*) password.....Set the password  
- (char \*) host.....Return the host in the url  
- host :(char \*) host ..... Set the host in the url  
- (int) port.....Return the port in the url  
- port :(int) port ..... Set the port in the url  
- (char \*) path ..... Return the path in the url  
- path :(char \*) path.....Set the path in the url  
- clear ..... Clear the url  
- (DText \*) url ..... Return the url  
- (BOOL) url :(char \*) url ..... Set the url  
- (BOOL) url :(char \*) url :(DURL \*) ref  
| Set url with reference url  
- (int) fromString :(char \*\*) cstr  
| Read url from string  
- (DText \*) toText ..... Convert url to text string