

Translating 3D Object

FIX OBJECT PIVOT objno	(773)	Realigns object's local axes
MOVE OBJECT objno, v	(772)	Moves object in the direction it faces by <i>v</i> units
MOVE OBJECT DOWN objno, v	(764)	Moves object down <i>v</i> units
MOVE OBJECT LEFT objno, v	(764)	Moves object left <i>v</i> units
MOVE OBJECT RIGHT objno, v	(764)	Moves object right <i>v</i> units
MOVE OBJECT UP objno, v	(764)	Moves object up <i>v</i> units
PITCH OBJECT UP objno, deg	(770)	Rotates object about x-axis by <i>deg</i> degrees
PITCH OBJECT DOWN objno, deg	(770)	Rotates object about x-axis by <i>-deg</i> degrees
POINT OBJECT objno, x, y, z	(771)	Points main polygon of object to (x,y,z)
POSITION OBJECT objno, x, y, z	(762)	Positions centre of object to (x,y,z)
ROLL OBJECT LEFT objno, deg	(771)	Rotates object about z-axis by <i>deg</i> degrees
ROLL OBJECT RIGHT objno, deg	(771)	Rotates object about z-axis by <i>-deg</i> degrees
ROTATE OBJECT objno, Xang, Yang, Zang	(768)	Rotates object to exact angle about all axes
SCALE OBJECT objno, Xsz, Ysz, Zsz	(775)	Resizes object as percentage in each dimension
SET OBJECT ROTATION XYZ objno	(769)	Sets order of multiple rotations to XYZ
SET OBJECT ROTATION ZYX objno	(769)	Sets order of multiple rotations to ZYX
TURN OBJECT LEFT objno, deg	(770)	Rotates object about y-axis by <i>deg</i> degrees
TURN OBJECT RIGHT objno, deg	(770)	Rotates object about y-axis by <i>-deg</i> degrees
XROTATE OBJECT objno, ang	(766)	Rotates object to exact angle <i>ang</i> about x-axis
YROTATE OBJECT objno, ang	(767)	Rotates object to exact angle <i>ang</i> about y-axis
ZROTATE OBJECT objno, ang	(767)	Rotates object to exact angle <i>ang</i> about z-axis

3D Object Appearance

COLOR OBJECT objno, colour	(795)	Changes colour of object to <i>colour</i>
FADE OBJECT objno, perc	(797)	Sets object's light absorption; <i>perc</i> of default
GHOST OBJECT OFF objno	(797)	Makes object opaque
GHOST OBJECT ON objno	(796)	Makes object transparent
HIDE OBJECT objno	(776)	Hides object
PLAY ANIMATION TO IMAGE fn, ino	(785)	Loads video <i>fn</i> to image <i>ino</i> (then use as texture)
SCALE OBJECT TEXTURE objno, usc, vsz	(782)	Tiles object texture; <i>usc</i> over U; <i>vsz</i> times over V
SCROLL OBJECT TEXTURE objno, x, y	(788)	Offsets the mapping of texture to object by <i>x, y</i>
SET DETAIL MAPPING ON objno,ino	(791)	Adds another image, <i>ino</i> , on top of texture
SET OBJECT CULL objno, mode	(784)	Sets culling of hidden faces (0 - off; 1 - on)
SET OBJECT FILTER objno, flfg	(793)	Modifies image filtering when texturing

<i>flfg</i>	0	-	no mipmap
	1	-	no smoothing
	2	-	linear filtering

Set smoothing of surface to *perc*%

Modifies texture mapping to object (*tm* 1 - normal; 2 - flipped; 3 - extend; 4 - once) (*mipflg* 0 - mipmap; 1 - no mipmap)

SET OBJECT SMOOTHING objno, perc	-	Sets smoothing of surface to <i>perc</i> %
SET OBJECT TEXTURE objno, tm, mipflg	(787)	Modifies texture mapping to object (<i>tm</i> 1 - normal; 2 - flipped; 3 - extend; 4 - once) (<i>mipflg</i> 0 - mipmap; 1 - no mipmap)
SET OBJECT TRANSPARENCY objno, tflg	(790)	Textures background areas (in)visible (1) 0
SET OBJECT WIREFRAME objno, mode	(783)	Sets object to wireframe mode (1) or normal (0)
SHOW OBJECT objno	(776)	Unhides objects
TEXTURE OBJECT objno, ino	(778)	Textures object using image <i>ino</i>

Cameras

AUTOCAM ON	(851)	Automatic camera positioning on
AUTOCAM OFF	(851)	Automatic camera positioning off
CAMERA ANGLE X{(camno)}:real	(844)	Returns camera's angle about x-axis
CAMERA ANGLE Y{(camno)}:real	(844)	Returns camera's angle about y-axis
CAMERA ANGLE Z{(camno)}:real	(844)	Returns camera's angle about z-axis
CAMERA POSITION X{(camno)}:real	(844)	Returns camera's x-coord
CAMERA POSITION Y{(camno)}:real	(844)	Returns camera's y-coord
CAMERA POSITION Z{(camno)}:real	(844)	Returns camera's z-coord
CLEAR CAMERA VIEW {camno.} colour	(869)	Clears camera viewport using colour
CONTROL CAMERA USING ARROWKEYS camno, spd, ang(856)		Camera moves <i>spd</i> units and turns <i>ang</i> degrees
DELETE CAMERA camno	(866)	Deletes camera <i>camno</i>
LOCK OBJECT OFF objno	(874)	Returns object to normal positioning
LOCK OBJECT ON objno	(874)	Fixes object position within camera's view
MAKE CAMERA camno	(863)	Creates a camera with ID <i>camno</i>
MOVE CAMERA {camno.} dist	(817)	Moves camera <i>dist</i> units
PITCH CAMERA DOWN {camno.} angle	(822)	Pitches camera down by <i>ang</i> degrees
PITCH CAMERA UP {camno.} angle	(822)	Pitches camera up by <i>ang</i> degrees
POINT CAMERA {camno.} X, Y, Z	(818)	Points camera at (x,y,z)
POSITION CAMERA {camno.} X, Y, Z	(816)	Positions camera at (x,y,z)
ROTATE CAMERA {camno.} Xang, Yang, Zang	(818)	Rotates camera to exact angles about all axes
ROLL CAMERA LEFT {camno.} angle	(823)	Tilts camera to left side by <i>ang</i> degrees
ROLL CAMERA RIGHT {camno.} angle	(823)	Tilts camera to right side by <i>ang</i> degrees
SET CAMERA ASPECT {camno.} ratio	(825)	Sets height/width distortion to <i>ratio</i>
SET CAMERA FOV {camno.}ang	(826)	Sets camera's field of view to <i>ang</i> degrees
SET CAMERA RANGE {camno.} min, max	(848)	Sets camera's near vision to <i>min</i> and far to <i>max</i>
SET CAMERA ROTATION XYZ	(820)	Sets the order of simultaneous rotations to <i>x,y,z</i>
SET CAMERA ROTATION ZYX	(820)	Sets the order of simultaneous rotations to <i>z,y,x</i>
SET CAMERA TO FOLLOW {camno.} x,y,z,ang,d,h,smth,coll(853)		Camera heads towards (x,y,z) . Stays <i>h</i> above at <i>d</i> away at angle of <i>ang</i> . Smoothness of movement <i>smth</i> (1 no smooth - 100 smooth)

SET CAMERA TO IMAGE camno, ino, w, h	(871)	<i>camno</i> 's output to image <i>ino</i> ; <i>w</i> width, <i>h</i> height
SET CAMERA TO OBJECT ORIENTATION {camno.}objno(873)		Camera points in same direction as <i>objno</i>
SET CAMERA VIEW {camno.} x1, y1, x2, y2	(845)	Sets screen coords of output to $(x1,y1),(x2,y2)$
SET CURRENT CAMERA camno	(865)	Sets current camera to <i>camno</i>
SET VECTOR3 TO CAMERA POSITION v3, camno	(875)	Camera's coords stored in 3D vector <i>v3</i>
SET VECTOR3 TO CAMERA ROTATION v3, camno	(876)	Camera's rotations stored in 3D vector <i>v3</i>
TURN CAMERA LEFT {camno.} angle	(822)	Turns camera left by <i>ang</i> degrees
TURN CAMERA RIGHT {camno.} angle	(822)	Turns camera right by <i>ang</i> degrees
XROTATE CAMERA {camno.} ang	(821)	Rotates camera to exact angle <i>ang</i> about x-axis
YROTATE CAMERA {camno.} ang	(821)	Rotates camera to exact angle <i>ang</i> about y-axis
ZROTATE CAMERA {camno.} ang	(821)	Rotates camera to exact angle <i>ang</i> about z-axis

Shading

SET CARTOON SHADING ON objno,shade,edge	(1054)	Shades object <i>objno</i> using images <i>shade</i> & <i>edge</i>
SET RAINBOW SHADING ON objno,ino	(1056)	Shades object <i>objno</i> using image <i>ino</i>
SET REFLECTION SHADING ON objno	(1057)	Turns <i>objno</i> into a mirrored surface
SET SHADING OFF objno	(1058)	Shading effects applied to <i>objno</i> switched off

Lights and Fog

COLOR AMBIENT LIGHT col	(889)	Sets the colour of the ambient light
COLOR LIGHT lghtno, col	(890)	Sets light <i>lghtno</i> to <i>col</i> colour
DELETE LIGHT lghtno	(890)	Deletes light <i>lghtno</i>
FOG COLOR col	(902)	Sets fog colour to <i>col</i>
FOG DISTANCE v	(902)	Sets distance at which fog obscures objects to <i>v</i>
FOG OFF	(901)	Hides fog effect on
FOG ON	(901)	Switches fog effect off
HIDE LIGHT lghtno	(887)	Hides the effect of light <i>lghtno</i>
LIGHT DIRECTION X{lghtno}:real	(900)	Returns x-coord of end point of 1 unit light beam
LIGHT DIRECTION Y{lghtno}:real	(900)	Returns y-coord of end point of 1 unit light beam
LIGHT DIRECTION Z{lghtno}:real	(900)	Returns z-coord of end point of 1 unit light beam
LIGHT EXIST{lghtno}:integer	(898)	Returns 1 if light <i>lghtno</i> exists
LIGHT POSITION X{lghtno}:real	(900)	Returns x-coord of light <i>lghtno</i>
LIGHT POSITION Y{lghtno}:real	(900)	Returns y-coord of light <i>lghtno</i>
LIGHT POSITION Z{lghtno}:real	(900)	Returns z-coord of light <i>lghtno</i>
LIGHT RANGE{lghtno}:real	(899)	Returns range of light
LIGHT TYPE{lghtno}:integer	(899)	Returns the light type (1 - direct, 2 - point, 3 - spot)
LIGHT VISBLE{lghtno}:integer	(899)	Returns 1 if light <i>lghtno</i> visible
MAKE LIGHT lghtno	(889)	Creates new light <i>lghtno</i>
POINT LIGHT lghtno, x,y,z	(893)	Points light <i>lghtno</i> in direction (x,y,z)
POSITION LIGHT lghtno, x,y,z	(891)	Positions light <i>lghtno</i> at (x,y,z)
ROTATE LIGHT lghtno, xan,yan,zan	(895)	Rotates light <i>lghtno</i> to <i>xan,yan,zan</i> about axes
SET AMBIENT LIGHT perc	(888)	Set ambient light density to <i>perc</i> %
SET DIRECTIONAL LIGHT lghtno, x,y,z	(892)	Changes light <i>lghtno</i> to directional placed at (x,y,z)
SET LIGHT RANGE lghtno, dist	(891)	Sets range of light <i>lghtno</i> to <i>dist</i> units
SET LIGHT TO OBJECT ORIENTATION lghtno, objno	(897)	Sets light to point in same direction as object <i>objno</i>
SET LIGHT TO OBJECT POSITION lghtno, objno	(895)	Sets light <i>lghtno</i> 's position to that of <i>objno</i>
SET POINT LIGHT lghtno, x,y,z	(893)	Changes light <i>lghtno</i> to a point light at (x,y,z)
SET SPOT LIGHT lghtno, ang1, ang2	(892)	Changes light <i>lghtno</i> to spot; inner <i>ang1</i> ;outer <i>ang2</i>
SET VECTOR3 TO LIGHT POSITION v3, lghtno	(904)	Stores light <i>lghtno</i> 's coords in 3D vector <i>v3</i>
SET VECTOR3 TO LIGHT ROTATION v3, lghtno	(904)	Stores light <i>lghtno</i> 's rotation angles in vector <i>v3</i>
SHOW LIGHT lghtno	(888)	Reshows a hidden light

Object Lighting

SET LIGHT MAPPING ON objno,ino	(1035)	Uses image <i>ino</i> as light map for object <i>objno</i>
SET GLOBAL SHADOW COLOR r,g,b,alpha	(1050)	Sets shadows colour to <i>r,g,b</i> transparency <i>alpha</i>
SET GLOBAL SHADOW SHADES v	(1050)	Sets number of shadow shades to <i>v</i>
SET GLOBAL SHADOWS OFF	(1048)	Sets off all shadows
SET GLOBAL SHADOWS ON	(1048)	Re-instates all shadows switched off
SET OBJECT AMBIENCE objno, col	(1029)	Sets ambient light from object <i>objno</i> to <i>col</i>
SET OBJECT DIFFUSE objno, col	(1030)	Sets diffuse light from object <i>objno</i> to <i>col</i>
SET OBJECT EMISSIVE objno, col	(1031)	Sets emissive light from object <i>objno</i> to <i>col</i>
SET OBJECT SPECULAR objno, col	(1030)	Sets specular light from object <i>objno</i> to <i>col</i>
SET OBJECT SPECULAR POWER objno, power	(1031)	Sets power of specular light from <i>objno</i> to <i>power</i>
SET SHADOW POSITION -1,x,y,z	(1051)	Sets position of light casting shadow to (x,y,z)
SET SHADOW SHADING OFF objno	(1048)	Sets off shadow shading for object <i>objno</i>
SET SHADOW SHADING ON objno {,mesh,dist,hard}	(1045)	Sets shadow on for <i>objno</i> , uses <i>mesh</i> as shadow up to <i>dist</i> units using hardware (<i>hard</i> = 0)

Meshes

CHANGE MESH objno, limbno, mshno	(931)	Changes the mesh used for an existing limb
DELETE MESH mshno	(915)	Deletes mesh <i>mshno</i>
LOAD MESH frame, mshno	(914)	Loads mesh <i>mshno</i> from file <i>fname</i>
MAKE MESH FROM OBJECT mshno, objno	(912)	Creates mesh <i>mshno</i> from object <i>objno</i>
MESH EXIST{mshno}:integer	(915)	Returns 1 if mesh <i>mshno</i> exists
SAVE MESH fname, mshno	(913)	Save mesh <i>mshno</i> to file <i>fname</i>

Limbs

CHECK LIMB LINK{objno, limbno}:integer	(947)	Returns 1 if link to <i>objno,limbno</i> is legal
LIMB ANGLE X{objno, limbno}:real	(939)	Returns limb's angle about local x-axis
LIMB ANGLE Y{objno, limbno}:real	(939)	Returns limb's angle about local y-axis
LIMB ANGLE Z{objno, limbno}:real	(939)	Returns limb's angle about local z-axis
LIMB DIRECTION X{objno, limbno}:real	(940)	Returns limb's angle about world x-axis
LIMB DIRECTION Y{objno, limbno}:real	(940)	Returns limb's angle about world y-axis
LIMB DIRECTION Z{objno, limbno}:real	(940)	Returns limb's angle about world z-axis
LIMB EXIST{objno, limbno}:integer	(936)	Returns 1 if limb <i>objno,limbno</i> exists
LIMB NAMES{objno, limbno}:string	(945)	Returns limb's name (given by 3D package)
LIMB OFFSET X{objno, limbno}:real	(937)	Returns limb's x-offset from centre of parent
LIMB OFFSET Y{objno, limbno}:real	(937)	Returns limb's z-offset from centre of parent
LIMB OFFSET Z{objno, limbno}:real	(937)	Returns limb's z-offset from centre of parent
LIMB POSITION X{objno, limbno}:real	(939)	Returns x-coord of limb's centre
LIMB POSITION Y{objno, limbno}:real	(939)	Returns y-coord of limb's centre
LIMB POSITION Z{objno, limbno}:real	(939)	Returns z-coord of limb's centre
LIMB SCALE X{objno, limbno}:real	(938)	Returns limb's width scaling (% of original)
LIMB SCALE Y{objno, limbno}:real	(938)	Returns limb's height scaling (% of original)
LIMB SCALE Z{objno, limbno}:real	(938)	Returns limb's depth scaling (% of original)
LIMB TEXTURE{objno, limbno}:integer	(946)	Returns ID of limb's texture image
LIMB TEXTURE NAME{objno, limbno}:string	(946)	Returns limb's texture name
LIMB VISBLE{objno, limbno}:integer	(937)	Returns 1 if limb <i>objno,limbno</i> is visible
MAKE OBJECT FROM LIMB obj1, obj2, limbno	(919)	Creates object <i>obj1</i> from limb <i>obj2,limbno</i>

BSP Files

LOAD BSP pk3file, bspfile	(1165)	Loads <i>bspfile</i> from archive <i>pk3file</i>
DELETE BSP	(1173)	Deletes the loaded BSP model
SET BSP CAMERA camno	(1173)	Makes camera <i>camno</i> a BSP camera
SET BSP CAMERA COLLISION cidx, camno, rad, resp	(1167)	Sets camera's collision radius and response
SET BSP OBJECT COLLISION cidx, objno, rad, resp	(1167)	Sets object's collision radius and response
SET BSP COLLISION THRESHOLD cidx, sens	(1171)	Sets <i>cidx</i> collision sensitivity
SET BSP COLLISION OFF cidx	(1171)	Switches off collision detection for <i>cidx</i>
SET BSP CAMERA COLLISION RADIUS cidx, camno, x,y,z(1169)		Reshapes camera's collision volume radii
SET BSP OBJECT COLLISION RADIUS cidx, objno, x,y,z(1169)		Reshapes object's collision volume radii
SET BSP COLLISION HEIGHT ADJUSTMENT cidx, h	(1170)	Adjusts offset of collision volume by <i>h</i>
SET BSP MULTITEXTURING ON	(1173)	Allows multitexturing of BSP model
SET BSP MULTITEXTURING OFF	(1173)	Disables multitexturing of BSP model
PROCESS BSP COLLISION cidx	(1171)	Forces check for collision involving <i>cidx</i>
BSP COLLISION HIT{cidx}:integer	(1172)	Returns 1 if <i>cidx</i> collision
BSP COLLISION X{cidx}:real	(1172)	Returns x-coord of collision involving <i>cidx</i>
BSP COLLISION Y{cidx}:real	(1172)	Returns y-coord of collision involving <i>cidx</i>
BSP COLLISION Z{cidx}:real	(1172)	Returns z-coord of collision involving <i>cidx</i>

Particles

COLOR PARTICLES pno, r, g, b	(1107)	Sets the colour of particles to <i>r,g,b</i>
DELETE PARTICLES pno	(1104)	Deletes particles <i>pno</i>
GHOST PARTICLES OFF pno, gmode	(1114)	Switches off ghosting for particles <i>pno</i>
GHOST PARTICLES ON pno, gmode	(1113)	Switch on ghosting for <i>pno</i> in mode <i>gmode</i>
HIDE PARTICLES pno	(1103)	Hides particles object <i>pno</i>
MAKE FIRE PARTICLES pno, ino, freq, x,y,z, w,h,d	(1119)	Creates fire (<i>w x h x d</i>) around point (x,y,z)
MAKE PARTICLES pno, ino, freq, rad	(1102)	Creates particles of size <i>rad</i> , frequency <i>freq</i>
MAKE SNOW PARTICLES pno, ino, freq, x,y,z,w,h,d	(1118)	Creates snow (<i>w x h x d</i>) around point (x,y,z)
PARTICLES EXIST{pno}:integer	(1114)	Returns 1 if particles object exists
PARTICLES POSITION X{pno}:real	(1115)	Returns x-coord of emitter
PARTICLES POSITION Y{pno}:real	(1115)	Returns y-coord of emitter
PARTICLES POSITION Z{pno}:real	(1115)	Returns z-coord of emitter
POSITION PARTICLE EMISSIONS pno, x,y,z	(1105)	Positions emitter to (x,y,z) old particles stay
POSITION PARTICLES pno, x,y,z	(1104)	Positions emitter to (x,y,z) ; old particles move
ROTATE PARTICLES pno, xang,yang,zang	(1106)	Rotates emitter to <i>xang,yang,zang</i>
SET PARTICLE CHAOS pno, chaos	(1110)	Sets random factor of particles to <i>chaos</i>
SET PARTICLE EMISSIONS pno, freq	(1108)	Sets the frequency of particles
SET PARTICLE FLOOR pno, floor	(1112)	If <i>floor</i> =0 no landing area for particles
SET PARTICLE GRAVITY pno, grav	(1110)	Sets the fall speed of the particles to <i>grav</i>
SET PARTICLE LIFE pno, perc	(1113)	Sets the lifetime of the particles to <i>perc</i>
SET PARTICLE SPEED pno, time	(1111)	Sets time gap between screen shots to <i>time</i>
SET PARTICLE VELOCITY pno, spd	(1113)	Sets speed of particles to <i>spd</i>
SET VECTOR3 TO PARTICLES POSITION v3 pno	(1116)	Saves emitter position in <i>v3</i>
SET VECTOR3 TO PARTICLES ROTATION v3, pno	(1116)	Saves emitter rotation in <i>v3</i>
SHOW PARTICLES pno	(1104)	Shows particles object <i>pno</i>

Matrices

DELETE MATRIX mno	-	Deletes matrix <i>mno</i>
FILL MATRIX mno, hght, tno	(1222)	Sets every square in matrix to height <i>hght</i> and textures it using image tile <i>tno</i>
GET GROUND HEIGHT(mno, x, z):real	(1218)	Returns height at coords (x,z)
GET MATRIX HEIGHT(mno, tx, tz):real	(1217)	Returns height at intersection (tx,tz)
GHOST MATRIX OFF mno	(1232)	Turns ghosting off
GHOST MATRIX ON mno	(1231)	Turns ghosting on
MAKE MATRIX mno, w, d, xs, zs	(1213)	Creates matrix <i>w x d</i> units in <i>xs x zs</i> segments
MATRIX EXIST(mno):integer	(1237)	Returns 1 if matrix <i>mno</i> exists
MATRIX POSITION X(mno):real	(1230)	Returns x coord of matrix bottom-left corner
MATRIX POSITION Y(mno):real	(1230)	Returns y coord of matrix bottom-left corner
MATRIX POSITION Z(mno):real	(1230)	Returns z coord of matrix bottom-left corner
MATRIX TILE COUNT(mno):integer	(1228)	Returns number of tiles in the prepared image
MATRIX TILES EXIST(mno):integer	(1228)	Returns 1 if an image has been prepared
MATRIX WIREFRAME STATE(mno):integer	(1220)	Returns 1 if matrix displayed in wireframe
POSITION MATRIX mno, X, Y, Z	(1219)	Positions bottom-left corner of matrix at (x,y,z)
PREPARE MATRIX TEXTURE mno, ino, vs, hs	(1220)	Prepare <i>ino</i> split into <i>vs x hs</i> tiles; used by <i>mno</i>
RANDOMIZE MATRIX mno, max	(1214)	Sets intersection heights to random up to <i>max</i>
SET MATRIX mno, wire, trans, cull, fill, lght, fog, amb	(1235)	Sets several characteristics in a single line
SET MATRIX HEIGHT mno, sx, sz, h	(1215)	Sets the height of intersection (sx,sz) to <i>h</i>
SET MATRIX NORMAL mno, sx, sz, nx, ny, nz	(1214)	Sets end of normal at (sx,sz) to (nx,ny,nz)
SET MATRIX PRIORITY mno, flag	(1232)	If <i>flag</i> =0, matrix is drawn first
SET MATRIX TILE mno, sx, sz, tno	(1223)	Tile <i>tno</i> from image placed in segment (sx,sz)
SET MATRIX TRIM mno, trimx, trimy	(1226)	Trims image tiles by <i>trimx</i> and <i>trimy</i>
SET MATRIX WIREFRAME OFF mno	(1219)	Displays matrix in solid mode
SET MATRIX WIREFRAME ON mno	(1219)	Displays matrix in wireframe mode
SET VECTOR3 TO MATRIX POSITION v3, mno	-	Records matrix position in vector <i>v3</i>
SHIFT MATRIX DOWN mno	(1227)	Shifts segments in matrix down 1 cell
SHIFT MATRIX LEFT mno	(1227)	Shifts segments in matrix left 1 cell
SHIFT MATRIX RIGHT mno	(1227)	Shifts segments in matrix right 1 cell
SHIFT MATRIX UP mno	(1227)	Shifts segments in matrix up 1 cell
UPDATE MATRIX mno	(1214)	Updates matrix display to show changes

Terrain

DELETE TERRAIN terno	(1181)	Deletes terrain <i>terno</i>
GET TERRAIN HEIGHT(terno, x,z):real	(1184)	Returns height of <i>terno</i> at position (x,z)
GET TOTAL TERRAIN HEIGHT(terno):real	(1186)	Returns highest point on <i>terno</i>
MAKE TERRAIN terno, file	(1180)	Creates terrain <i>terno</i> using file for <i>height</i>
POSITION TERRAIN terno, x,y,z	(1182)	Positions top-left corner of <i>terno</i> at (x,y,z)
TERRAIN EXIST{terno}:integer	-	
TERRAIN POSITION X(terno):real	(1183)	Returns x-ord <i>terno</i> 's top-left corner
TERRAIN POSITION Y(terno):real	(1183)	Returns y-ord <i>terno</i> 's top-left corner
TERRAIN POSITION Z(terno):real	(1183)	Returns z-ord <i>terno</i> 's top-left corner
TEXTURE TERRAIN terno, ino	(1183)	Textures <i>terno</i> using image <i>ino</i>

Advanced Terrain

BUILD TERRAIN terno,flag	(1188)	Makes <i>terno</i> visible, <i>flag</i> 0 = smoothing
GET TERRAIN GROUND HEIGHT (terno,x,z):real	(1191)	Returns height at position (x,z)
GET TERRAIN SIZE X (terno):real	(1193)	Returns the width of the terrain object
GET TERRAIN SIZE Z (terno):real	(1193)	Returns the depth of the terrain object
LOAD TERRAIN fname,terno	(1194)	Loads terrain from file <i>fname</i>
MAK OBJECT TERRAIN terno	(1186)	Creates terrain object <i>terno</i>
SAVE TERRAIN fname,terno	(1193)	Saves terrain to file <i>fname</i>
SET TERRAIN HEIGHTMAP terno,file	(1187)	Uses file to contour <i>terno</i>
SET TERRAIN LIGHT terno,x,y,z,r,g,b,flag	(1190)	Light at (x,y,z) colour(<i>r,g,b</i>) <i>flag</i> 0=dark 1=light
SET TERRAIN SCALE terno,xs,ys,zs	(1187)	Scales <i>terno</i> into <i>xs x ys</i> factor along x-axis, etc.
SET TERRAIN SPLIT terno,spd	(1191)	Splits terrain by <i>spd</i> <i>x</i> <i>sp</i> meshes
SET TERRAIN TEXTURE terno, grain, txt	(1188)	Sets terrain <i>terno</i> with

DarkBASIC Pro

Statements

for the book

Hands On DarkBASIC Pro

Volume 2

Book titles available:

Hands On Pascal
Hands On C++
Hands On Java
Hands On XHTML
Hands On DarkBASIC Pro 1 & 2

Creating Basic 3D Objects

CLONE OBJECT objno1, objno2	(777)	Creates an independent copy of <i>objno2</i>
DELETE OBJECT objno	(777)	Deletes object <i>objno</i>
INSTANCE OBJECT objno1, objno2	(778)	Creates a dependent copy of <i>objno2</i>
MAKE OBJECT BOX objno, w, h, d	(757)	Creates a box <i>objno</i> of dimensions <i>w,h,d</i>
MAKE OBJECT CONE objno, sz	(760)	Creates a cone <i>objno</i> ; height and diameter <i>sz</i>
MAKE OBJECT CUBE objno, sz	(756)	Creates a cube <i>objno</i> ; dimensions <i>sz</i>
MAKE OBJECT CYLINDER objno, sz	(759)	Creates a cylinder <i>objno</i> ; height and diameter <i>sz</i>
MAKE OBJECT PLAIN objno, w, h	(760)	Creates 2D plain <i>objno</i> ; width <i>w</i> and height <i>h</i>
MAKE OBJECT SPHERE objno, sz {,row,col}	(758)	Creates a sphere <i>objno</i> ; diameter <i>sz</i> ; increase polygon count using <i>row</i> and <i>col</i>
MAKE OBJECT TRIANGLE objno, x1, y1, z1, x2, y2, z2, x3, y3, z3	(761)	Creates a triangle <i>objno</i> with vertices at (x1,y1,z1),(x2,y2,z2),(x3,y3,z3)
PERFORM CSG UNION obj1, obj2	(787)	Object <i>obj1</i> modified to union of <i>obj1</i> and <i>obj2</i>
PERFORM CSG DIFFERENCE obj1, obj2	(789)	Object <i>obj1</i> modified to difference of <i>obj1</i> and <i>obj2</i>
PERFORM CSG INTERSECTION obj1, obj2	(789)	Object <i>obj1</i> modified by overlap of <i>obj2</i>
SET GLOBAL OBJECT CREATION mode	(785)	mode=1 objects share vertex buffer

Loading/Saving 3D Objects

LOAD OBJECT file, objno	(950)	Loads object <i>objno</i> from file
MAKE OBJECT objno, meshno, ino	(914)	Creates <i>objno</i> from <i>meshno</i> textured with <i>ino</i>
SAVE OBJECT file,objno	(949)	Saves <i>objno</i> to file (DBO format)

Animated 3D Objects

APPEND OBJECT file, objno, frame	(970)	Adds frames in <i>file</i> to <i>objno</i> . Insert at <i>frame</i>
PLAY OBJECT objno, st,fin	(965)	Plays <i>objno</i> starting at <i>st</i> , finishing at <i>fin</i>
LOOP OBJECT objno, st,fin	(966)	Plays <i>objno</i> continuously between <i>st</i> and <i>fin</i>
STOP OBJECT objno	(968)	Stops <i>objno</i> playing
SET OBJECT FRAME objno, frame,flag	(968)	Sets <i>objno</i> to <i>frame</i> . <i>flag</i> =1; recalc bounding box
SET OBJECT SPEED objno, spd	(967)	Set <i>objno</i> play speed to <i>spd</i> (1 slow - 100 fast)
SET OBJECT INTERPOLATION objno, ipol	(969)	Set <i>objno</i> 's frame jump (1 smooth - 100 instant)
OBJECT PLAYING(objno):integer	(971)	Returns 1 if <i>objno</i> playing; else 0
OBJECT LOOPING(objno):integer	(971)	Returns 1 if <i>objno</i> looping; else 0
OBJECT FRAME(objno):integer	(972)	Returns current frame of <i>objno</i>
OBJECT SPEED(objno):integer	(972)	Returns speed setting for <i>objno</i>
OBJECT INTERPOLATION(objno):integer	(972)	Returns interpolation setting for <i>objno</i>
TOTAL OBJECT FRAMES(objno):integer	(966)	Returns total frames in <i>objno</i>

3D Object Data

OBJECT ANGLE X(objno):real	(781)	Returns angle of rotation about local x-axis
OBJECT ANGLE Y(objno):real	(781)	Returns angle of rotation about local y-axis
OBJECT ANGLE Z(objno):real	(781)	Returns angle of rotation about local z-axis
OBJECT EXIST(objno):integer	(779)	Returns 1 if object exists
OBJECT POSITION X(objno):real	(779)	Returns x-ord of object
OBJECT POSITION Y(objno):real	(779)	Returns y-ord of object
OBJECT POSITION Z(objno):real	(779)	Returns z-ord of object
OBJECT SIZE X(objno):real	(780)	Returns width of object
OBJECT SIZE Y(objno):real	(780)	Returns height of object
OBJECT SIZE Z(objno):real	(780)	Returns depth of object
OBJECT VISIBLE(objno):real	(780)	Returns 1 if object is visible

Effects

DELETE EFFECT fxno	(1381)	Deletes effect <i>fxno</i>
EFFECT EXIST (fxno):integer	(1378)	Returns 1 if effect <i>fxno</i> exists
LOAD EFFECT frame,fxno,txtflg	(1378)	Loads effect <i>fxno</i> from <i>frame</i> ; <i>txtflg</i> =1 - textured using images specified in shader
PERFORM CHECKLIST FOR EFFECT ERRORS	(1379)	Creates checklist for errors in effect
PERFORM CHECKLIST FOR EFFECT VALUES	(1382)	Creates checklist of values used in effect
SET EFFECT CONSTANT type fxno,name,value	(1383)	Assigns <i>value</i> to parameter <i>name</i> in <i>fxno</i>
SET EFFECT TECHNIQUE fxno,name	(1383)	Specifies <i>name</i> of technique used in <i>fxno</i>
SET EFFECT TRANSPOSE fxno,flag	(1384)	If <i>flag</i> =1 matrices for <i>fxno</i> transposed
SET EFFECT ON objno,frame,txtflg	(1380)	Object <i>objno</i> has effect in <i>frame</i> applied; <i>txtflg</i> =1 using images specified in shader
SET LIMB EFFECT objno,limbno,fxno	(1381)	Effect <i>fxno</i> applied to limb <i>objno</i> , <i>limbno</i>
SET OBJECT EFFECT objno,fxno	(1379)	Effect <i>fxno</i> applied to object <i>objno</i>

Regular Collisions

AUTOMATIC OBJECT COLLISION objno,rad,resp	(1082)	Deletes <i>objno</i> 's collision box
DELETE OBJECT COLLISION BOX objno	(1082)	Returns sliding collision data x offset
GET OBJECT COLLISION X():real	(1080)	Returns sliding collision data y offset
GET OBJECT COLLISION Y():real	(1080)	Returns sliding collision data z offset
GET OBJECT COLLISION Z():real	(1080)	Hides bounding volumes of <i>objno</i>
HIDE OBJECT BOUNDS objno	(1072)	Returns shortest distance between line (x1,y1,z1) to (x2,y2,z2) and <i>objno</i>
INTERSECT OBJECT(objno, x1,y1,z1,x2,y2,z2):real	(1083)	

MAKE OBJECT COLLISION BOX objno,x1,y1,z1,x2,y2,z2,rotate (1077)

OBJECT COLLISION(objno1, objno2):integer	(1070)	Creates bounding box for <i>objno</i> . Corners (x1,y1,z1)(x2,y2,z2) rotates if <i>rotate</i> =1
OBJECT COLLISION CENTER X(objno)	(1075)	As OBJECT HIT() below
OBJECT COLLISION CENTER Y(objno)	(1075)	Returns x coord of collision volume of <i>objno</i>
OBJECT COLLISION CENTER Z(objno)	(1075)	Returns y coord of collision volume of <i>objno</i>
OBJECT COLLISION CENTER X(objno)	(1075)	Returns z coord of collision volume of <i>objno</i>
OBJECT COLLISION RADIUS(objno):real	(1075)	Returns collision radius of <i>objno</i>
OBJECT HIT (objno,0):integer	(1069)	Returns ID of object <i>objno</i> hits; 0 if no hit
OBJECT HIT(objno1, objno2):integer	(1069)	Returns 1 if <i>objno1</i> has hit <i>objno2</i>
SHOW OBJECT BOUNDS objno	(1072)	Displays bounding volumes of <i>objno</i>
SET GLOBAL COLLISION OFF	(1071)	Detects no collisions
SET GLOBAL COLLISION ON	(1071)	Re-instates collision detection
SET OBJECT COLLISION OFF objno	(1070)	Switches off collision detection for <i>objno</i>
SET OBJECT COLLISION ON objno	(1070)	Switches on collision detection for <i>objno</i>
SET OBJECT COLLISION TO BOXES	(1076)	Uses bounding box when detecting collision
SET OBJECT COLLISION TO POLYGONS objno	(1076)	Uses bounding polygons when detecting collision
SET OBJECT COLLISION TO SPHERES objno	(1074)	Uses bounding sphere when detecting collision
SET OBJECT RADIUS objno,rad	(1074)	Sets <i>objno</i> 's bounding sphere radius to <i>rad</i>

Static Collisions

GET STATIC COLLISION HIT(ox1, oy1, oz1, ox2, oy2, oz2, nx1, ny1, nz1, nx2, ny2, nz2):integer	(1087)	Returns 1 if leading box intersects collision box
GET STATIC COLLISION X():real	(1089)	Returns penetration distance x component
GET STATIC COLLISION Y():real	(1089)	Returns penetration distance y component
GET STATIC COLLISION Z():real	(1089)	Returns penetration distance z component
MAKE STATIC COLLISION BOX x1,y1,z1,x2,y2,z2	(1087)	Creates collision box enclosing given volume
STATIC LINE OF SIGHT(x1, y1, z1, x2, y2, z2, w, acc):integer	(1093)	Returns 1 if line (width <i>w</i>) meets collision box
STATIC LINE OF SIGHT X():real	(1095)	Returns x-coord of line's contact with box
STATIC LINE OF SIGHT Y():real	(1095)	Returns y-coord of line's contact with box
STATIC LINE OF SIGHT Z():real	(1095)	Returns z-coord of line's contact with box

Memory Blocks

CHANGE MESH FROM MEMBLOCK mshno,mno	(1310)	Updates mesh <i>mshno</i> using block <i>mno</i>
COPY MEMBLOCK mno1, mno2,off1, off2,bytes	(1287)	Copies <i>bytes</i> from <i>off1</i> in <i>mno1</i> to <i>off2</i> in <i>mno2</i>
DELETE MEMBLOCK mno	(1286)	Deletes block <i>mno</i>
GET MEMBLOCK PTR (mno):integer	(1281)	Returns start address of block <i>mno</i>
GET MEMBLOCK SIZE(mno):integer	(1286)	Returns size of block <i>mno</i>
MAKE BITMAP FROM MEMBLOCK bno,mno	(1302)	Makes bitmap <i>bno</i> using block <i>mno</i>
MAKE FILE FROM MEMBLOCK fno,mno	(1292)	Writes block <i>mno</i> to file <i>fno</i>
MAKE IMAGE FROM MEMBLOCK ino,mno	(1302)	Makes image <i>ino</i> using block <i>mno</i>
MAKE SOUND FROM MEMBLOCK mno	(1305)	Makes sound <i>sno</i> using block <i>mno</i>
MAKE MEMBLOCK mno,bytes	(1281)	Creates block <i>mno</i> containing <i>bytes</i> bytes
MAKE MEMBLOCK FROM BITMAP mno,bno	(1297)	Creates block <i>mno</i> using bitmap <i>bno</i>
MAKE MEMBLOCK FROM FILE mno,fno	(1294)	Creates block <i>mno</i> using file <i>fno</i>
MAKE MEMBLOCK FROM IMAGE mno,ino	(1302)	Creates block <i>mno</i> using image <i>ino</i>
MAKE MEMBLOCK FROM MESH mno,mshno	(1306)	Creates block <i>mno</i> using mesh <i>mshno</i>
MAKE MEMBLOCK FROM SOUND mno,sno	(1303)	Creates block <i>mno</i> using sound <i>sno</i>
MAKE MESH FROM MEMBLOCK mshno,mno	(1309)	Creates mesh <i>mshno</i> from block <i>mno</i>
MEMBLOCK value(mno,p):value	(1285)	Returns data, <i>type value</i> from byte <i>p</i> in <i>mno</i>
MEMBLOCK EXIST(mno):integer	(1286)	Returns 1 if block <i>mno</i> exists
READ MEMBLOCK frame,mno	(1292)	Reads file <i>frame</i> into block <i>mno</i>
WRITE MEMBLOCK value mno,p,v	(1284)	Writes data <i>v</i> of <i>type value</i> to byte <i>p</i> in <i>mno</i>
WRITE MEMBLOCK frame,mno	(1290)	Writes block <i>mno</i> to file <i>frame</i>

Vertex Buffers

ADD MESH TO VERTEXDATA meshno	(1265)	Adds data from <i>meshno</i> to buffer
DELETE MESH FROM VERTEXDATA vs,vf,is,if	(1271)	Deletes elements vs to <i>vf</i> and is to <i>if</i> in buffer
GET INDEXDATA(post):integer	(1268)	Returns contents of poly(<i>post</i>)
GET VERTEXDATA U(vno):real	(1255)	Returns U offset for vertex vert(<i>vno</i>)
GET VERTEXDATA V(vno):real	(1255)	Returns V offset for vertex vert(<i>vno</i>)
GET VERTEXDATA DIFFUSE(vno):integer	(1258)	Returns diffuse value for vertex vert(<i>vno</i>)
GET VERTEXDATA INDEX COUNT():integer	(1267)	Returns no. of entries in polygon array
GET VERTEXDATA NORMALS X(vno):real	(1253)	Returns x-coord of normal for vertex <i>vno</i>
GET VERTEXDATA NORMALS Y(vno):real	(1253)	Returns y-coord of normal for vertex <i>vno</i>
GET VERTEXDATA NORMALS Z(vno):real	(1253)	Returns z-coord of normal for vertex <i>vno</i>
GET VERTEXDATA POSITION X(vno):real	(1248)	Returns x-coord of vertex <i>vno</i>
GET VERTEXDATA POSITION Y(vno):real	(1248)	Returns y-coord of vertex <i>vno</i>
GET VERTEXDATA POSITION Z(vno):real	(1248)	Returns z-coord of vertex <i>vno</i>
GET VERTEXDATA VERTEX COUNT	(1247)	Returns no. of entries in vert array
LOCK VERTEXDATA FOR LIMB objno,limbno,mode	(1251)	Captures data of <i>objno</i> 's <i>limbno</i> . <i>mode</i> 0,1,2
LOCK VERTEXDATA FOR MESH meshno,mode	(1246)	Captures data of <i>meshno</i> . <i>mode</i> options 0,1,2

SET INDEXDATA pno,vno	(1270)
SET VERTEXDATA DIFFUSE vno,col	(1257)
SET VERTEXDATA NORMALS vno,x,y,z	(1254)
SET VERTEXDATA POSITION vno,x,y,z	(1250)
SET VERTEXDATA UV vno,Uoff,Voff	(1256)
UNLOCK VERTEXDATA	(1250)

Open Dynamics Engine

ODE ADD FORCE objno,xf,yf,zf,x,y,z	(1338)	Adds forces <i>fx,fy,fx</i> at point (x,y,z) to object <i>objno</i>
ODE COLLISION GET MESSAGE	(1336)	Retrieves top message on collision stack
ODE COLLISION MESSAGE EXISTS()	(1336)	Returns 1 if a message is in the collision stack
ODE CREATE DYNAMIC BOX objno,w,h,d	(1318)	Applies collision box (size <i>w x h x d</i>) to <i>objno</i>
ODE CREATE DYNAMIC CYLINDER objno	(1322)	Applies collision cylinder to <i>objno</i>
ODE CREATE DYNAMIC SPHERE	(1322)	Applies collision sphere to <i>objno</i>
ODE CREATE DYNAMIC TRIANGLE MESH objno	(1324)	Applies collision mesh to <i>objno</i>
ODE CREATE STATIC BOX objno	(1321)	Creates static box about <i>objno</i>
ODE CREATE STATIC TRIANGLE MESH objno	(1325)	Creates static mesh about <i>objno</i>
ODE DESTROY OBJECT objno	(1334)	Destroys ODE object linked to 3D <i>objno</i>
ODE END	(1319)	Shuts down ODE
ODE GET BODY HEIGHT (objno):real	(1335)	Returns height of <i>objno</i> at current rotation
ODE GET BODY LINEAR VELOCITY X (objno):real	(1335)	Returns the x component of object <i>objno</i> 's velocity
ODE GET BODY LINEAR VELOCITY Y (objno):real	(1335)	Returns the y component of object <i>objno</i> 's velocity
ODE GET BODY LINEAR VELOCITY Z (objno):real	(1335)	Returns the z component of object <i>objno</i> 's velocity
ODE GET OBJECT A():integer	(1336)	Returns ID of first object in a collision
ODE GET OBJECT B():integer	(1336)	Returns ID of second object in a collision
ODE GET OBJECT (A/B) ANGULAR VELOCITY X():real	(1338)	Returns angular velocity of collision obj about x-axis
ODE GET OBJECT (A/B) ANGULAR VELOCITY Y():real	(1338)	Returns angular velocity of collision obj about y-axis
ODE GET OBJECT (A/B) ANGULAR VELOCITY Z():real	(1338)	Returns angular velocity of collision obj about z-axis
ODE GET OBJECT (A/B) VELOCITY X():real	(1337)	Returns velocity of collision object along x-axis
ODE GET OBJECT (A/B) VELOCITY Y():real	(1337)	Returns velocity of collision object along y-axis
ODE GET OBJECT (A/B) VELOCITY Z():real	(1337)	Returns velocity of collision object along z-axis
ODE SET CONTACT FDIR objno,frict	(1328)	Sets contact friction to <i>frict</i>
ODE SET ANGULAR VELOCITY objno,xc,yc,zc	(1331)	Sets the angular velocities for each axis to <i>xc,yc,zc</i>
ODE SET BODY MASS objno,v	(1332)	Sets body's mass to <i>v</i>
ODE SET BODY ROTATION objno, xa,ya,za	(1332)	Sets object's rotation for each axis to <i>xa,ya,za</i>
ODE SET LINEAR VELOCITY objno,xc,yc,zc	(1328)	Sets object's linear velocity to <i>xc,yc,zc</i>
ODE SET WORLD CFM v	(1327)	Sets constrain force mixing to <i>v</i>
ODE SET WORLD ERP v	(1326)	Sets error reduction parameter to <i>v</i>
ODE SET WORLD GRAVITY xc,yc,zc	(1320)	Sets gravitation pull along each axis to <i>xc,yc,zc</i>
ODE SET WORLD STEP time	(1325)	Sets world time between calculations to <i>time</i>
ODE START	(1319)	Starts up ODE
ODE UPDATE	(1319)	Updates all ODE calculations

Networking

CREATE NET GAME game, player, max, setup	(1394)	Session <i>game</i> hosted by <i>player</i> with <i>max</i> users
CREATE NET PLAYER(player):integer	(1412)	setup=1 - peer-to-peer; setup=2 - client/server
FREE NET GAME	(1411)	Pseudo user <i>player</i> created ; returns ID assigned
FREE NET PLAYER id	(1412)	Removes user from current session
GET NET MESSAGE	(1401)	Removes pseudo <i>player</i> id from session
JOIN NET GAME session,player	(1396)	Makes first arrived message available
NET BUFFER SIZE():integer	(1408)	<i>player</i> joins session
NET GAME EXISTS(x):integer	(1413)	Returns number of messages in buffer
NET GAME LOST():integer	(1413)	Returns 1 if client/server host exists
NET GAME NOW HOSTING	(1411)	Returns 1 if host no longer exists
NET MESSAGE type():value	(1402)	Returns 1 if the current user becomes host
NET MESSAGE EXISTS():integer	(1402)	Returns <i>value</i> of next message which is of <i>type</i>
NET MESSAGE PLAYER FROM():integer	(1403)	Returns 1 if a message has arrived
NET MESSAGE PLAYER TO():integer	(1403)	Returns local ID of message sender
NET MESSAGE TYPE():integer	(1406)	Returns value indicating type of current message
NET PLAYER CREATED():integer	(1409)	Returns local ID of user joining session
NET PLAYER DESTROYED():integer	(1409)	Returns local ID of user leaving session
PERFORM CHECKLIST FOR NET CONNECTIONS	(1391)	Creates checklist of possible connections
PERFORM CHECKLIST FOR NET PLAYERS	(1397)	Creates checklist of current session's users
PERFORM CHECKLIST FOR NET SESSIONS	(1395)	Creates checklist of sessions available
SEND NET MESSAGE type, player, value	(1401)	Sends <i>value</i> of <i>type</i> to <i>player</i>
SET NET CONNECTION type, address	(1392)	Create a connection of <i>type</i> using <i>address</i>

File Transfer Protocol

FTP CONNECT url,user,pass	(1436)	Creates connection to <i>url</i> giving <i>user</i> name and <i>pass</i> password
FTP DELETE file	(1442)	Deletes file from FTP site
FTP DISCONNECT {disconnect}	(1440)	Disconnects using <i>disconnect</i> for dial-ups
FTP FIND FIRST	(1438)	Creates a reference to first file in directory
FTP FIND NEXT	(1439)	Moves reference to next file in directory
FTP GET FILE file {,newfile {,bytes}}	(1440)	Downloads file, saves as <i>newfile</i> , <i>bytes</i> in each packet
FTP PROCEED	(1441)	Downloads next packet of file
FTP PUT FILE file	(1443)	Uploads file to FTP site
FTP SET DIR dir	(1438)	Sets current FTP site directory to <i>dir</i>
FTP TERMINATE	(1442)	Terminates current file download
GET FTP DIRS():string	(1438)	Returns current FTP site directory
GET FTP ERRORS():string	(1437)	Returns description of last FTP failure
GET FTP FAILURES():integer	(1437)	Returns # if last FTP statement failed
GET FTP FILE NAMES():string	(1439)	Returns name of currently referenced file
GET FTP FILE SIZE():integer	(1439)	Returns the bytes in current file
GET FTP FILE TYPE():integer	(1439)	0 = file; 1 = directory; -1 = no more files
GET FTP PROGRESS():integer	(1442)	Returns % of file downloaded (-1 =100%)
GET FTP STATUS():integer	(1437)	Returns 0 if connection failed

DDLs

CALL DLL dllno,funcnt,params	(1463)	Runs <i>funcnt</i> in <i>dllno</i> using <i>params</i> parameters
DELETE DLL dllno	(1467)	Deletes <i>dllno</i>
DLL CALL EXIST (dllno,funcnt):integer	(1464)	Returns 1 if <i>funcnt</i> in <i>dllno</i> ; else 0
DLL EXIST (dllno):integer	(1463)	Returns 1 if <i>dllno</i> was loaded
LOAD DLL frame,dllno	(1463)	Loads DLL from <i>frame</i> gives ID <i>dllno</i>

poly(<i>pno</i>) set to <i>vno</i>	
Diffuse for vertex <i>vno</i> set to <i>col</i> colour	
Sets end of vertex <i>vno</i> 's normal to (x,y,z)	
Set vertex <i>vno</i> coords to (x,y,z)	
Sets vertex <i>vno</i> texture offsets to <i>Uoff,Voff</i>	
Updates object whose data held in vertex buffer	