□Part A

Microsoft(DirectX(3 *ƒƒƒƒƒƒ*Š”*ƒƒƒ*

™ ™ ,*ƒƒƒƒƒƒ*,‹□,,,,,□•,□— □,,,•□,,,,,,,,□,,□•’,‰Ž–□–

,,,, “‹“ ‹Š“,,,,,,•Ž •–,,,• “’,,,,,,,,,
Microsoft , ,,*ƒƒƒƒƒƒ*,‹ ,,,,, *ƒƒƒƒ* *ƒƒƒ*,“‹,,,, ’,“‹ • ’ Œ ,,‘,’“ —
Œ,•Ž,,,, ,,*ƒƒƒƒƒƒ*, Microsoft , –,,,‹‰,,,, “‹ • ’ Œ ,,‘,’“ —Œ,‹‰,—
,,,,,,,,

‘5
Direct3D,Š—
µDirect3D,Š— §
Direct3D,,,, §
Direct3Dƒ ƒƒƒƒƒ §
3Dƒƒƒƒƒƒƒ,Š‘‹ §
ƒƒƒ ƒƒƒ, “‰ §
•Žƒ ƒ,Š— §
•Žƒ ƒ,,,, §
ƒƒƒƒƒƒƒƒƒ: Direct3D,•Žƒ ƒ ƒƒƒƒƒƒ §

Direct3D,•Žƒ□ƒ,ƒƒ□ƒƒƒƒ §

’□ƒ□ƒ,Š—..§
’□ƒ ƒ,,,,..§
ƒƒƒƒƒƒƒƒƒ: Direct3D,’ ƒ ƒ ƒƒƒƒƒƒ..§
Direct3D,’ ƒ ƒ,ƒƒ ƒƒƒƒ..§

Direct3D,Š—

Direct3D„„

Microsoft,Š'3Dƒƒƒƒƒƒƒ ƒƒƒ ƒƒƒ,, Direct3D™,,,OpenGL,ƒƒƒƒ ƒƒƒ ƒ ƒƒƒƒƒƒ ƒƒƒ ƒƒƒƒ API ,Š,,,,,

Ž, , Windows,ƒƒƒƒƒƒƒ ƒƒƒƒ, Win32®

API,‹ ,,,ƒƒƒƒ□ƒƒƒ□„'‚Windowsƒƒƒƒ□„„ƒ□ƒƒƒƒ,,ŠŒ,Ž,,,,,,,

.... µ §

Direct3D

Direct3D, Œ Ž—,,,,,,ƒ ƒƒƒƒƒƒƒ ƒ PC ,ƒƒƒƒƒƒ,ƒƒƒ ƒƒƒ ,Ž—

,,ƒ ƒ„„Microsoft,'‹„□ƒƒƒƒƒƒ,ƒƒƒƒƒƒƒƒ,3D‹ ,,, Direct3D, '" ,'ˆ,ƒƒƒƒ, ,,,,

Direct3D, Š"Ž,•—

,,,APIƒ ƒƒ,ƒƒƒƒ"ˆ' ,'‹, ƒ ƒƒƒƒ ƒƒƒ ,,ˆ"",ƒƒƒƒ ƒƒƒ,‹‹,,□Direct3D„„ □ƒ ƒƒƒ ƒƒƒƒ ƒ, 'Œ,3DƒƒƒƒƒƒŠ‹,,,,ƒƒ ƒ,,,,,,, ƒƒƒ ƒ ƒ,ƒƒƒƒ,ƒƒƒƒƒ ,3DƒƒƒƒƒƒŠ‹,'‰,,,,,Š',,, ƒƒƒƒ,,,,,ƒƒƒ,,•—

,,,,,, ƒƒƒƒ ƒ ,,,„„ƒƒƒƒ□ƒ□ƒƒƒƒ□,‰",“ ,,

Direct3D,ƒƒƒ ƒƒƒ 3D

ƒƒƒƒƒƒƒ ƒ ƒƒ,Š'ƒƒƒ,,, ƒƒƒƒƒƒ ƒ ƒ, ',ƒƒƒƒƒƒ ƒƒƒƒƒƒ •Š – ƒƒƒ —

, ƒ ƒƒƒƒ ƒƒƒƒƒ ƒ,,' ",ƒƒƒƒ,ŽŒ„□APIƒ□ƒƒ„□"ˆ„□ƒƒƒ,•Žƒ□ƒAPI,□'ƒ ƒƒ,'□ƒ□ƒAPI„□Direct3D,Ž—

„3Dƒ ƒƒƒƒ ƒƒƒƒƒ ƒ,ƒƒƒƒ,,ƒƒƒƒ,ƒƒ ƒ,, Direct3D,Š',ƒƒ ƒƒƒ, ƒ ƒƒƒƒ, Direct3D,,,,3Dƒƒƒƒƒƒ ƒƒƒƒƒƒ,ˆ•,,,'•, '‰

,,,,,,,, Direct3D,□Žƒƒƒƒƒƒƒ□ƒƒƒƒƒƒƒƒƒƒ□ƒƒƒƒ□ƒƒƒƒƒƒƒ□ƒƒƒƒƒƒƒƒ ƒƒƒƒƒ Š,ƒ ƒ,,,ƒƒƒƒƒ ƒƒƒƒƒ,,, 3Dƒ ƒƒƒƒ ƒƒƒƒƒ ƒ,Š'ƒƒƒƒƒƒƒ"—

, ,,,,,,,, Direct3D,',DirectXƒƒƒƒƒ ,Š',“ ,,,,,,, ƒƒƒ ƒƒƒƒƒ 2D,ƒ□ƒ□ƒƒ □ƒƒ□ƒ„3Dƒƒƒƒƒƒ ƒƒƒƒƒ,,,,Š'‹",“ ,,,, ,,,,,, ƒƒƒƒƒƒƒƒ,ƒƒƒƒ ƒƒƒƒ, ƒ ƒ ƒƒ,2D,3D,ƒƒƒƒƒƒƒ,Ž—,,,,,,,,

Direct3D,ƒƒƒƒƒƒƒ•–

,, •Žƒ ƒ,' ƒ□ƒ„„□„„,ˆ„ „„,ƒ ƒ,,, •Žƒ ƒ, ƒƒƒƒ ƒƒƒ,ƒƒƒƒƒƒ ƒ ƒ,•Ž,, ƒƒƒAPI,,, ' ƒ ƒ , ƒƒƒƒ ƒƒƒ,–Ž",Ž ƒƒƒƒ,ƒ ƒ, —,,'ƒƒƒAPI,,,

,, ,, Direct3D,' ƒ ƒ,•Žƒ□ƒ„„□–„„„„□Direct3D,'„——

„„„„„ƒ□ƒƒƒƒ□ƒƒƒƒƒƒƒƒƒ□'□‰□,ƒƒƒƒ ƒƒƒ,,,,,Ž,ˆ,

•..•Žƒ ƒ

•..' ƒ ƒ

•..ƒ□ƒƒƒƒ□ƒƒƒƒƒƒƒƒƒ□'□‰□,ƒƒƒƒ ƒƒƒ

•Žƒ ƒ

Direct3D,•Žƒ□ƒAPI,□3Dƒƒƒƒƒƒ,' ,3Dƒƒƒƒƒƒƒ,□—,,,,□Œ,,,,,□•Žƒ□ƒ,— ,,,□Š', Windowsƒƒƒƒ□ƒƒƒ,,3D‹",'‰, ,,3Dƒƒƒƒ□ƒƒƒ,□□,— ˆ,□„„„„□•Žƒ□ƒ,',□„„„ƒƒƒƒƒ□ƒƒƒƒ,□ƒ□ƒƒ□ƒ□ƒƒƒ□のののの

の 3D 1 API のの

の DirectDraw® API

Microsoft のの Windows

DirectDraw : Direct3D の

Direct3D のMicrosoft の 3D API
の',ƒƒ□ƒƒƒ□ƒƒƒ‚ƒƒƒƒƒƒ Windows

の 3D ののの

の

ののの3D のの

Direct3D OLE COM
DirectDraw Microsoft のの Windows Direct3D
: Direct3D の

Direct3D API の DirectX API
HAL HAL のDirect3D HAL
HEL Direct3D
HEL の
のAPI

Direct3D HAL DirectDraw HAL GDI
HAL Microsoft の API の3D のDirect3D DirectDraw GDI
OpenGL の 3D の 3D のDirect3D HAL
のの

DirectDraw

DirectDraw
DirectDraw 2D 3D のの Windows
DirectDraw の
の DirectDraw の
のの Windows の

DirectDraw
DirectDraw Windows
ののDirect3D DirectDraw
Microsoft の MS-DOS®
DirectDraw DirectDraw の

DirectDraw COM の API Microsoft のの Windows DirectDraw
DirectDraw

OpenGL

OpenGL CAD/CAM 3D の3D のOpenGL
WindowsNT Windows95 Windows95 OpenGL
Win32 Win32 OpenGL
の OpenGL OpenGL

OpenGL の

の Direct3D,•Š,,ƒ ƒƒƒƒ ƒƒƒ ƒƒƒ,'‹, Direct3D API,',,'‰Š,3Dƒ ƒƒƒƒ OpenGL,•—,,, –, Œ,ƒƒ ƒ,,,, ,—— ,,,,,,,,—',,,

Direct3Dƒ ƒƒƒƒƒ

,, ,, Direct3Dƒƒƒ ƒƒƒ,,,',DirectXƒƒƒ ƒƒƒ,ƒƒƒ ƒƒƒƒ ƒƒƒƒ ,,,ƒƒƒƒ ƒ ƒ ƒƒƒ,,ŠŒ,,,,,' ", •,‹ ,,,,, ,,,,ˆ‰,ƒƒƒƒ,,,, –,,

- *Direct3D,ƒƒƒƒ*
-
-
-
-
-
-
-
- Direct3D

Direct3D の

Direct3D Windows
3D Direct3D
3D の

3D
の
Direct3D の
Direct3D の □□□□□□□□□ DirectDraw DirectDraw
3D のの
の 2D の
の 3D の

Direct3D 3D
□□□□□□□□□□ の Direct3D
3D *の*
のののの
の Direct3Dƒƒƒƒ□ƒƒƒ,□

Direct3D DirectDraw DirectDraw の COM
Direct3D の COM
Direct3D DirectDraw の DirectDraw
の 3D の3D の

のののDirect3D のののののの

Direct3D の 3D

- の 4 4 ののののののの
-
- のの

の

2D 3D の

ののののの Direct3D

§

DirectDraw Direct3D

のの DirectDraw

§

”„,□„□Direct3D,–,□„,□ƒƒƒƒ□ƒƒƒ□ƒ□ƒ,“□„„ƒƒƒ□ƒƒƒ,Š

Direct3D

ののののDirect3D

ISV の

のの Direct3D API の

Direct3D のD3DOPCODE のD3DINSTRUCTION ののの

§

ののののD3DOP_TRAIANGLE

のの

のののの

IDirect3DDevice::GetCaps D3DDEVICEDESC の

dwMaxBufferSize Ž—

„„,□□□ˆ”“,□□“,Ž□ƒƒƒ 64K のの

Direct3D のののの

の 4 の
のXの
3D のXXの
3D 2D

のの

のの

の D3DTLVERTEX D3DTLVERTEX

D3DLVERTEX のの
D3DLVERTEX

D3DVERTEX のののののの

D3DTRANSFORMDATA

のののの API
のの*3D*

ƒƒƒ„„„ƒƒƒƒ‚•Ž‚

D3DLIGHTINGELEMENT
の
のXの

RGB のD3DLIGHTDATA の
D3DLIGHTINGELEMENT の

の” ”
のXの RGB のXの
の

の0の 1ののの

§

のののののののの 3/4 ののの 4 の 1 ののの□□ □□,□

ƒƒƒ,,,RGB*ƒƒ*□,□Ž,,,,’‹,,,,,□

```
#define RGB_MAKE (red, green, blue) \
      ((red)   << 16) | \
      ((green) << 8)  | \
      (blue))
```

RGBA の

```
#define RGBA_MAKE(red, green, blue, alpha) \
     (((alpha) << 24) | \
      ((red)   << 16) | \
      ((green) << 8)  | \
      (blue))
```

Direct3D, , Ž,,,,’‹,,,,,

```
typedef unsigned long D3DCOLOR;
```

ŒŒ,ƒƒƒ, D3DLIGHTTYPE

—‹*Œ,ƒƒƒ*,,,, ,,,, D3DLIGHT_DIRECTIONAL D3DLIGHT_POINT D 3DLIGHT_PARALLELPOINT D3D*LIGHT_SPOT D3DLIGHT_GL*SPOT, ,,,,,,,,,,,, ,,—‹Œ, D3DLIGHT ‘‘,ˆ•,,, ,, ‘‘,, ‘,D3DCOLORVALUE ‘‘,Š,,,,, *ŒŒ, ,Ž’,,,,* — ,—‘,’,”ˆ, ˜”,0,,1,,,,, *ƒƒƒ*ŒŒ*ƒƒƒ*,’, Ž,“Ž,Š,,,,,

§

Š—‘,,0,,1,”ˆŠ,’,Ž’‰”, *ƒ ƒ ƒƒƒ*,,,, –Œ‰ ,Š’,,,,,,,, *D3DLIGHT* ‘‘,•Ž,,•Œ*ƒƒƒƒ*, *ƒƒƒ*,,ŒŒ,,,•Œ,•,,,, ,,*ƒƒƒƒ*, — ŒŒŒ,,,,• ,–,,,,,,,,, *ƒƒƒƒ*,,,,*ƒ ƒƒ* •,*Ž’*,, ,,,,*ƒƒƒƒ*, Œ ,*ƒ ƒƒ* — ,,,,*ƒƒƒ* •,•Š,,, ,,,, *ƒƒƒƒ*,*ƒ ƒƒ* •,•Š,,,,,, Œ—“,*ƒƒƒ*, –Œ‰ ,Ž,,,,,,, *ƒƒƒƒƒƒƒ*,*ƒƒƒƒƒƒƒ*,, ”ˆ*ƒƒƒ ƒ*, — Œ,ŒŒ,”ˆ,Ž,,,, ,,”ˆ,Š•,,,’“,, Œ,“,,,,,,,,, Œ,‹,,2ŽŒ Œ ,,,,•*‰* ,, *Œ,“,,,,,’“,,ŒŒ,,,‹—,d,,,,* Ž,•’Ž, —,,

§

D3DL*IGHT*□*‘‘,Ž,,ƒƒƒ*□*dvTheta,,,dvPhi*□*,*□*,,,,*□*ƒƒƒƒƒƒƒ*,—„‰ ,Œ□,Š“,’‹„□Œ□Œ□□dvFalloff

の
の D3DLIGHTDATA

のの

IDirect3DDevice::Execute ののの

のの(0, 0)の (−1, −1)の 0 1
のののののののののの

Direct3D の

のDirect3D のの

-
-

IDirect3DRM::CreateDeviceFromSurface DirectDraw
Direct3D
DirectDraw の DiretDraw
の SDK のDirect3D
WM_ACTIVATE
Direct3D 8 の DirectDraw の
IDirectDrawPalette::GetEntries

ののの

D3DPAL_FREE

D3DPAL_READONLY

D3DPAL_RESERVED

のWin32 の PALETTEENTRY の peFlags のD3DRMPALETTEENTRY の D3DRMPALETTEFLAGS RGB

の

RGB の

D*3DTLVERTEX* の *specular* の

RGBF ”F” ”fog”の F

のの

の

の

の

の

§

••Ž□*ƒƒƒ*□*ƒ*□*ƒ*„□Ž□*ƒƒƒ*□*ƒ*

§

の e 2.71828,,, *ƒƒƒ*,“–“,•,Š ,,, *ƒƒƒ*,’, ,,,, *ƒƒƒƒƒƒ*,Œ,,,,,,

ƒƒƒƒ ƒƒƒ,Ž *ƒƒƒ ƒ ƒ*,Ž—,, *ƒƒƒ*,–“,0.5,,,,,,, Ž,—,Ž,,,, *ƒƒƒ*,,0.8,‹— ,,,ˆ’,*ƒƒƒ*,’,0.6703,,,

§

ƒƒ ƒ,ƒƒ ƒ ƒ ƒ ƒƒƒ

Direct3D, Œ , ‘, ’,,,,,, Ž *ƒƒƒƒ*’,*ƒ ƒ*,‰ ,, *ƒƒƒƒ ƒƒƒ*, *ƒƒƒƒ*,*ƒ ƒ*,*ƒƒ ƒƒƒƒ*,,,,,ŽŽ,,‘, ‘,*ƒƒƒƒƒƒ*,, D3DSTATE ‘‘,, ,,*ƒ ƒƒƒƒƒ*,Ž ,,Ž,,,,—‹Œ, Š,,,,, D3DTRANSFORMSTATETYPE,•Š*ƒƒƒ ƒ*, ‘, ’,, ,, D3DLIGHT STATETYPE, –*ƒƒƒ ƒ*, ‘, *D3DRENDERSTA*TETYPE,*ƒƒƒ* — *ƒƒƒ ƒ*, ‘, ’,,

,,,,, ‘, “,Ž, —,Ž,BOOL’,*ƒƒƒ*,Ž,,,, ,,*ƒƒƒ*,TRUE, ,,ˆ ,• , ,,,,

ƒƒƒƒ ƒƒƒ, D3DSTATE_OVERRIDE*ƒƒƒ*,Ž—,, “,Ž, — ‘,– Œ,,,,,,,,, ,,‹”,,,,*ƒƒƒƒ ƒƒƒ*, Ž *ƒƒƒƒ*, —— , *ƒƒƒƒ*, ‘,•,,,,,,,“ ,• ,,,,,,,, *Dir*ect3D,•Ž*ƒ ƒ*,, *ƒƒ ƒ ƒ ƒ ƒƒƒ*,— —,,Ž ,,,*ƒƒƒ*,,, *ƒƒ ƒ ƒ ƒ ƒƒƒ*,—,,,, Ž *ƒƒƒƒ*,Š‘, ’,,•— , ,,,,, ,,,, *ƒƒƒƒ*,*ƒƒƒƒƒ*,*ƒƒ ƒ*,*ƒƒƒƒƒ*,’,Š,,,, •Ž*ƒ ƒ*API,*ƒƒ ƒ ƒ ƒ ƒƒƒ* ,——,,

ƒƒƒƒ□*ƒƒƒ*,□

ƒƒ□□*ƒƒ*□*ƒƒƒƒ*□*ƒ*□*ƒ*,*ƒƒƒ*,,,*ƒƒƒƒƒ*,,,,,,D3DSTATE_OVERRIDE*ƒƒƒ*,Ž— ,,,,,,,,□,,の

D3DRENDERSTATETYPE の D3DRENDERSTATE_SHADEMODE

OP_STATE_RENDER(2, lpBuffer);

STATE_DATA(D3DRENDERSTATE_SHADEMODE, D3DSHADE_GOURAUD, lpBuffer);

STATE_DATA(D3DSTATE_OVERRIDE(D3DRENDERSTATE_SHADEMO
DE), TRUE, lpBuffer);

OP_STATE_RENDER D3DOP_STATERENDER
D3DOP_STATERENDER D3DOPCODE のの
D3DSHADE_GOURAUD D3DSHADEMODE の
の

1 3DSTATE_OVERRIDE

STATE_DATA(D3DSTATE_OVERRIDE(D3DRENDERSTATE_SHADEMO
DE), FALSE, lpBuffer);

OP_STATE_RENDER STATE_DATA DirectX SDK の Misc
D3dmacs.h のStep 5: のの

Direct3D

Direct3D の 3D のののののの

Direct3D の.x の DirectX™ SDK Autodesk 3D
Studio .3ds Direct3D のの.xof
Conv3ds.exe Convxof.exe

Direct3D の API Direct3D

3D の

の3D ののののののDirect3D の ƒƒ,Ž„„

3D Direct3D のの

3D

3D の 2 の の
のx の y の z
の のx y

のの
のの z の

の Direct3D の

- Direct3D の
- U V

Direct3D の

Direct3D のの z の

§

のの

Direct3D

-

v0 **v1** **v2** v0 v2 v1 Direct3D

- Z −1 の D3DMATRIX の_13
_33 _43 の

U V

Direct3D のu v
v のz u [0,0,0]
y u v IDirect3DRMWrap

3D

3D の

- の の
- の
-

のの4 4 のの(x, y, z) (x’, y’, z’)

§

の(x’, y’, z’) (x, y, z)

§

の

§

のの

```
D3DMATRIX scale = {
   D3DVAL(s),    0,            0,            0,
   0,            D3DVAL(s),    D3DVAL(t),    0,
   0,            0,            D3DVAL(s),    D3DVAL(v),
   0,            0,            0,            D3DVAL(1)
};
```

のDirect3D 3D

•

•

•

のののののの API
のDirect3D

の(x, y, z) (x’, y’, z’)

§

の □,“,,,,,,□‘,ƒƒƒ

の(x, y, z) x (x’, y’, z’)

§

のy

§

のz

§

のののθ

の(x, y, z) x y z の(x’, y’, z’)

§

Direct3D の 3 のののの

の **Di**rect3D

-
-
-
- の

§

のの の

§

の の
Direct3D のの Direct3D

§

Direct3D ののの

§

ののの
の
ののの
の
の

Note

のの

§

ののの

ƒƒƒƒƒƒƒ‚‘□„„„‚‚

の

§

§

の

のの

□□□□□□□ の

のDirect3D

の

のの

•

•

•

•

の

のののの

のの

ののの

のRGB D3DCOLOR_RGB
D3DCOLOR_MONO

1 ののの 0.8 2 のの 0.4 の のののの
□□□の RGB 0.6

の
□□□□□□ のの

□□□□□□ の □□□□□□
D3DPRIMCAPS の dwShadeCaps

の

ののの
□□□□□□ 7

§

の v0□v1□v2 □□□□□□2 の v1□v3□v2 □□3 □□□□□□□□ v3□v4□v2 □□□□□2 ののの
□‰,,Œ,,•‰,,,,,,,,,□
□Œ,‟Š□,Ž,,,,□,,,,ŽŠŒ,,,,,’‟,‹—,,‟,ˆ,,,,,□
§
ƒƒƒƒ,□’‟v0□v1□v2,—,,□□,ŽŠŒ,•‰,,□2″–,ŽŠŒ,•‰,,□’‟v0□v2□v3,Ž—,,□
‟Š□,□Œ D3DTRIANGLE の wFlags

□□□□□□□□□□□ƒƒƒƒ
Direct3D,,□’‟,ˆ’,Œ,,Ž,,,,□ƒƒƒƒƒƒ’,Š’‟,□ˆ’,Ž,ƒƒƒƒ□Œ,,Ž,–
□ƒƒƒƒ□ƒƒƒƒƒ,□•□,,,□,,,,‹□,,,□•Žƒ□ƒ,,□,,,,’,D3DRMVERTEX□‘‘,Š”,,,□□

ƒƒ□ƒƒƒƒ,□ƒƒƒƒ,’‹,,[x, y, z]’,4,,,—�second,’‰,,□,,,□3D‰“,ˆ”“,—,,,,,,,□–
ƒƒƒƒ,‘,,,,,,,,□ƒƒ□ƒƒƒƒ,□3D‹Š,Ž,□,,Ž,‰,,‰“,•,,,,,□,,,,,ƒƒ□ƒ □□□□□□(1, 1, 2)
□1 □□□□□の□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□,,,,□,,□‰
,□□,•Š,,,,,,,,‘□,Ž□,,,,,”Š,,,□
ƒƒ□ƒƒƒƒ,□□,□ƒƒ□ƒƒƒƒ,Œ□,,,,,,,, の

q1 o q2 **2** 2 1 1

q1 q2 の Q = q1 o q2

Q q2 q1 1

のの

の

2 r2 の 1 の r2

の

□□□

Direct3D ののの D3DRMQuaternionFromRotation

D3DRMQUATERNION

D3DRMQuaternionMultiply

D3DRMQuaternionSlerp

D3DRMQuaternionFromRotation

D3DRMQuaternionMultiply

D3DRMQuaternionSlerp

D3DRMVectorAdd

D3DRMVectorCrossProduct

D3DRMVectorDotProduct

D3DRMVectorModulus

D3DRMVectorNormalize

D3DRMVectorRandom

D3DRMVectorReflect

D3DRMVectorRotate

D3DRMVectorScale

D3DRMVectorSubtract

Direct3D の DirectX 53 の
の 53 のの

の

3D

の

の

の の の の Direct3D

- の
  - の
  - の
  -

Direct3D の
RGB の の

- の
-
- Z
-

IDirect3DDevice::Execute *ƒƒƒƒ*,—
,,,,,*ƒƒƒƒƒƒ*,,,

の

の

IDirect3DDevice::Execute の

D3DSTATUS の
の D3DOPCODE の
D3DOP_BRANCHFORWARD
Direct3D の
の

の

のDirect3D
HAL
HAL

の

のののの

- CPU の
- の
- の 256×256 の4 の 128×128 の256×256 の 1 ののの256×256 の,,,,□,,,,□‘

D3DTRIANGLE の wFlags のの

のの *ƒƒƒƒ ƒƒƒ*,•–,Ž ,,,,,,,,
D3DTRIFLAG_STARTFLAT(len)

Œ ,ŽŠŒ,Ž, ,,,, *ƒƒƒƒ*, *“Š* , *Œ*,, *Œ*,‘,*ŽŠŒ*,len,Ž, ,,Ž, ,,,,,,,

D3DTRIFLAG_ODD,D3DTRIFLAG_EVEN

ƒƒƒƒ,□ŽŠŒ,,,,,,,□,,’“,,, *ƒ*□*ƒ*,,,,,□‘,,,,,’“,*ƒƒƒƒƒ*

の**D3DTRIFLAG_STARTFLAT** **D3DTRIFLAG_ODD**
D3DTRIFLAG_EVEN の

D3DTRIFLAG_STARTFLAT のの
の
の

D3DTRIFLAG_ODD D3DTRIFLAG_EVEN
D3DTRIFLAG_START
D3DTRIFLAG_START の
D3DTRIFLAG_START の D3DTRIFLAG_ODD
D3DTRIFLAG_EVEN

の SDK D3DTRIFLAG_ODD D3DTRIFLAG_EVEN

の

の

のの

- の
- の64
-

ƒƒƒƒ‚ƒƒƒ ƒƒƒ‚16ˆ‰‚‚‚

-

‚‚‚‚ƒƒƒƒƒ ƒƒƒƒƒ ƒƒƒƒ‚ ‚ ‚“ˆ‚
‚ ƒƒƒƒƒ‚ƒƒ ƒƒƒ‚Ž’‚‚‚‚‚‚‚‚ ‚‚‚‚
‚‚‚‚ƒƒƒƒƒ‚”‚‚‚ ”Ž—
‚“‚‚•‚ ‘‚‚ƒƒƒƒ ƒƒƒ‚‚ ‚‚‚‚ƒ ƒ‚‚‚
‚ˆ ‚ƒƒƒƒƒ‚•—
‚‚ ‚‚‚‚Œ‚”Ž‚‚‚‚ƒƒƒƒƒƒ‚‚‚ ‚‚‚‚‚‚”

Ž‚

-
- の256×256 の 1 の
-

のの
□□□□□□□ のの

のののの
□□□□□□□□□ のののの
ƒ□ƒ□ƒƒ‚Ž
8 の
のののの

Z

Z
Z
Z
の

Z

の 2
Z

Z Z
の
の

API Z ののの Z

のの SDK の D3dtest.exe の4 の

のののの

の SDK の Direct3D の
のの¥ののD3dtest.exe ¥のののの

Direct3D のの-systemmemory
ののfail
DirectDraw WIN16
WIN16ƒƒƒ‚GDI‚USER„ƒƒƒƒ‚ƒƒƒƒ‰‚IDirectDrawSurface2::Lock
IDirectDrawSurface2::Unlock のの Windows
IDirectDrawSurface2::GetDC IDirectDrawSurface2::ReleaseDC のの

D3DTEXTUREBLEND の
D3DTBLEND_COPY

8 の
16
16
の 8 のの 2
8 ののの
D3d**test.exe**

の

のDirect3D の 3D Microsoft
3D Direct3D の API

Direct3D DirectDraw DirectDraw
DirectDraw Direct3D ののIDirectDraw::QueryInterface

DirectDraw の IDirect3D Direct3D
DirectDraw のDirect3D

の 3D ののの
3D : Direct3D の
3D の 3D
の **SDK**
の

の3D のMicrosoft □ 3D □□□□□□□□ □□□□□ □□□□□□□□□□□**3** □□□の□□□□□□□□□□□□□□□□□□□□の□□□□□
□□□□□□□□□□□□□□□□□□□□□**Direct3D** ののの□□

```
□ƒƒƒƒƒ,,,,,□□Direct3D,•Žƒ□ƒ,ƒƒ□ƒƒƒƒ□,Ž□,,,,□
ƒƒƒƒƒƒƒƒƒ: Direct3D,•Žƒ□ƒ□ƒƒƒƒƒƒ
Direct3D,•Žƒ□ƒ,,ƒƒƒƒ,□,,,,□,ƒƒƒƒƒƒ,’,,□,,,□ˆ‰
,•,,,,,ƒƒƒƒƒƒ,ƒƒƒƒƒƒ,□,,,,,,,,Š’,□–,,□
ƒƒƒƒƒƒ
□–

Direct3DRMAnimation
Ž,Direct3DRMFrameƒƒƒƒƒƒ,Ž□,,,,,□,,,,,•Š,Ž,‰
,,,,’‹,,ƒƒƒƒƒƒ,,,□,,ƒƒƒƒƒƒ,□Direct3DRMVisual□Direct3DRMLight□Direct3D
RMViewportƒƒƒƒƒƒ,ˆ’,Œ,□ƒƒ□ƒ,ƒƒƒ□ƒƒƒ,,,,,Ž,,,,,,,□
```

Direct3DRMAnimationSet

Direct3DRMAnimatio**n**

Direct3DRMDevice

のの

Direct3DRMFace

のの

Direct3DRMFrame

の の

Direct3**DRMLight**

5 の のの

Direct3DRMMaterial

のの

Direct3DRMMesh

のXの

Direct3DRMMeshBuilder

```
□□□□□□□□□□□□□□□□ƒƒ,,,Œ□,’“,–,‘□,,,,,,,,□

Direct3DRMObject
Direct3D,‘,,,,,•Žƒ□ƒ□ƒƒƒƒƒƒ,—,,Š–ƒƒƒ,,,□,,,,ƒƒƒƒƒƒ,‹’,“’,•Ž,,,,□

Direct3DRMPickedArray
,,ƒƒƒƒƒƒ,□—,,,,2D,“,‘‰,,ƒƒƒ
```

Direct3DRMShadow

Direct3DRMTexture

の

Direct3DRMUserVisual

Direct3*DRMViewport*

の3D の 2D

Direct3DRMVisual

ƒ,ƒƒƒƒƒƒ,,,,,,,,ƒƒƒƒƒƒ,,,,□ƒƒƒƒƒ□ƒƒƒƒƒƒ,‰

Direct3DRMWrap

のの

の COM GetElement GetSize のののIDirect3DRM

IObjectName::QueryInterface のののIDirect3DRMDevice::QueryInterface
IDirect3DRMWinDevice
IDirect3DRMVisual

Direct3DRMAnimation

IDirect3DRMAnimation

Direct3DRMAnimationSet

IDirect3DRMAnimationSet

Direct3DRMDevice

IDirect3DRMDevice, IDirect3DRMWinDevice

Direct3DRMFace

IDirect3DRMFace

Direct3DRMFrame

IDirect3DRMFrame, IDirect3DRMVisual

Direct3DRMLight

IDirect3DRMLight

Direct3DRMMaterial

IDirect3DRMMaterial

Direct3DRMMesh

IDirect3DRMMesh, IDirect3DRMVisual

Direct3DRMMeshBuilder

IDirect3DRMMeshBuilder, IDirect3DRMVisual

Direct3DRMShadow

IDirect3DRMShadow, IDirect3DRMVisual

Direct3DRMTexture

IDirect3DRMTexture, IDirect3DRMVisual

Direct3DRMUserVisual

IDirect3DRMUserVisual, IDirect3DRMVisual

Direct3DRMViewport

IDirect3DRMViewport

Direct3DRMWrap

IDirect3DRMWrap

の Direct3DRMDevice
IDirect3DRM::CreateObject Direct3DRMDevice
の
IDirect3DRMDevice::InitFromClipper
*IDirect3DRMDevice::QueryInterf*ace のDirect3DRMDevice ののWM_PAINT
WM_ACTIVATE の IDirect3DRMWinDevice

```
d3drmapi->CreateObject(CLSID_CDirect3DRMDevice, NULL,
   IID_IDirect3DRMDevice,(LPVOID FAR*)&dev1);
dev1->InitFromClipper(lpDDClipper, IID_IDirect3DRMDevice,
   r.right, r.bottom);
```

```
dev1->QueryInterface(IID_IDirect3DRMWinDevice, (LPVOID*) &dev2);
```

QueryInterface
の

Direct3D IDirect3DRMObject IUnknown
IDirect3DRMObject
•—
,,,,, ƒƒƒŽ•Ž CLSID ,Ž,,,,, IDirect3DRM::CreateObjectƒƒƒƒ,Œ, ,,, ”
—ƒƒƒƒƒƒ, ,,,,,,,,, ,,‘,, Šƒƒƒ ƒƒƒƒ,,,,ˆ‰, ƒƒƒƒ,Ž—,,
”—ƒƒƒ ƒƒƒƒ
ƒƒƒƒ

IDirect3DRMDeviceArray

IDir*ect3DRM::GetDe*vices

IDirect3DRMFaceArray
IDirect3DRMMeshBuilder::GetFaces

IDirect3DRMFrameArray

IDirect3DRMPickedArray::GetPick

IDirect3DRMFrame::GetChildren

IDirect3DRMLightArray

IDirect3DRMFrame::GetLights

IDirect3DRMPickedArray

IDirect3DRMViewport::Pick

IDirect3DRMViewportArray

IDirect3DRM::CreateFrame

IDirect3DRMVisualArray

IDirect3DRMFrame::GetVisuals

ƒƒƒƒƒ‚ Ž ƒƒƒƒ‚0‚‚‚‚ ‚‚‚‚‚‚‚

ƒƒƒƒ ƒƒƒ‚ ƒ ƒ‚ƒ ƒƒƒƒƒƒƒ‚‚‚‚‚ Ž ƒƒƒƒ‚•Ž‚‚•—
‚‚‚ ‘‚ƒƒƒƒƒƒ‚Ž ƒƒƒƒ‚ Ž“"‚ƒƒƒƒ‚Š—‚‚ ƒƒƒƒ ƒƒƒ‚ —
‚‚‘‚ƒ ƒ‚ƒƒ ƒ ƒ ƒƒƒƒ‚‰•‚‚‚‚‚‚ ƒƒƒƒ ƒƒƒ‚ƒƒ ƒ ƒ‚‰•‚‚‚ ƒƒƒƒ‚Ž“"‚ƒƒ
ƒ‚Ž ƒƒƒƒ‚ —‚‚ —˜"‚‚ ‚‚ƒƒ ƒ ƒ‚ƒƒƒƒ‚'‰
‚‚ ‚‚‚‚ ƒƒƒƒ ƒƒƒ‚ƒƒƒƒ‚‰•‚‚‚ƒƒ ƒ ƒ‚‰•‚‚‚‚‚‚‚‚ ‚‚‚ ƒƒƒƒ‚‰•‚‚‚‚‚ "
—‚ƒƒ ƒ ƒ‚‰•‚‚‚‚‚‚‚‚‚

Žƒƒƒƒƒƒ‚ƒƒƒƒƒ□ƒƒƒƒƒƒ‚Ž□ƒƒƒƒ‚□ƒƒ□ƒ‚'‰ ‚‚‚‚‚‘‚‚‚□IDirect3DRMFrame::AddChildƒƒƒ‚Ž— „□„□ƒƒƒƒƒƒ„•‚□‚Žƒƒƒƒƒƒ‚ˆ"„‚□ƒƒƒƒ‚Ž"“‚Ž□ƒƒƒƒ‚‘□„□
ƒƒƒƒ□ƒƒƒ‚ƒƒƒƒƒ

ののののの

IDirect3DRM

IDirect3DRM の
Direct3DRM のののIdirect3DRM

IDirect3DRM のの

- ƒƒƒ□ƒƒ
-
- face
- ƒƒ ƒ
- Š‰,,,,,,ˆ”ƒƒƒƒƒƒ
- Œ
- ƒƒƒƒƒ
- ƒƒƒƒ,ƒƒƒƒ ƒƒƒ
- ‰‰
- ƒƒƒƒƒ
- ƒ ƒ ƒƒƒƒƒ
-
-

IDirect3DRMAnimation IDirect3DRMAnimationSet

Direct3DRMAnimation

Direct3DRMFrame

Direct3DRMAnimation Direct3DRMVisual
Direct3DRMLight Direct3DRMViewport

IDirect3DRMAnimation::AddPositionKey
IDirect3DRMAnimation::AddRotateKey
IDirect3DRMAnimation::AddScaleKey のの 99 の 49 のの0
のの

IDirect3DRMAnimation::SetTime ののの
IDirect3DRMAnimation::SetTime の

Direct3DRMAnimationSet □□□□□□□Direct3DRMAnimationSet □□□□□□□□□□□□□□□□□
のの ƒƒƒ□ƒ,‹—
,□•Ž,ƒƒƒ□ƒƒƒ,□□,Š—‰
,,,□ƒƒƒƒ□ƒƒƒ,□IDirect3DRMAnimationSet::AddAnimationƒƒƒƒ,—
,,ƒƒƒ□ƒƒƒ,ƒƒƒ□ƒƒƒ□ƒƒƒ,’‰
,,,,,,,,□ƒƒƒ□ƒƒƒ,□□,,,,□IDirect3DRMAnimationSet::DeleteAnimationƒƒƒƒ,
Ž—,,□ƒƒƒ□ƒƒƒ□ƒƒƒ,□IDirect3DRMAnimationSet::SetTimeƒƒƒƒ,Œ,□,,,,,□Œ,,,□

Š˜□•,,,,,□□IDirect3DRMAnimation□,,,□IDirect3DRMAnimationSet□ƒƒƒ□ƒƒƒƒ,
Ž□,,,,□
IDirect3DRMDevice,IDirect3DRMDeviceArrayƒƒƒ□ƒƒƒƒ
ƒƒƒƒƒƒ,,,,,,,□—ŒŽ,□□–ƒƒƒƒ,Š˜,,,,,,,,,,,□ƒƒƒƒ□ƒƒƒƒƒƒ,□ƒƒƒƒƒƒ,□—
□,‰Žƒƒƒƒƒƒ,•,,,,□
ƒƒƒƒƒƒ,‶□,□Ž′,,,□

,,,

•Žƒ ƒ, ƒƒƒ ƒ,ƒƒƒƒƒ ,,,ƒƒƒƒ ƒƒƒ ƒƒƒ,’ ƒƒƒƒƒƒ,,ƒƒƒƒ,ƒƒ ƒ,,

Š˜ •,,,,, *IDi*rect3DRMDevice ,Ž ,,,,

,,ƒƒƒƒƒ,, Direct3Dƒƒ ƒ, —ƒƒƒƒ,•Ž,, ,—Œ,ƒƒƒƒƒ,,,, –,,

- ƒƒƒƒƒ
- ƒƒ ƒƒƒ
-

IDirect3DRMDevice::SetQuality
IDirect3DRMMesh*Buil*der::SetQuality のの IDirect3DRMDevice::GetQuality
IDirect3DRMMeshBuilder::GetQuality

RGB

IDirect3DRMDevice::GetColo*rModel*

RGB ,•□,ŒŒ
の 8 16 24
32 のの
24 のの

の

の □ƒ,,,,のののの RGB のの
の
の
RGB 8 16 24 32 の
8 の
RGB

Direct3D のIDirect3D::EnumDevices IDirect3D::FindDevice のの

WM_MOVE
WM_PAINT WM_ACTIVATE
IDirect3DRMWinDevice::HandlePaint
IDirect3DRMWinDevice::HandleActivate Direct3D

IDirect3DRMWinDevice

IDirect3DRMFace IDirect3DRMFaceArray

のの
IDirect3DRMFace::SetColor
IDirect3DRMFace::SetColorRGB□IDirect3DRMFace::SetTexture
IDirect3DRMFace::SetMaterial の

IDirect3DRMFace::AddVertex
IDirect3DRMFace::AddVertexAndNormalIndexed
の
IDirect3DRMFace::GetVertices IDirect3DRMFace::GetVertexƒƒƒƒ‚Ž—

IDirect3DRMFace

IDirect3DRMFrame IDirect3DRMFrameArray

のののののの

ののの
の□□□の□□□□□□ の

のの 1 NULL IDirect3DRM::CreateFrame

*Direc*t3D の3D

IDirect3DRMFrame

のの

-
-
-
-

の

の　　　　　　　　　　　　のの の

IDirect3DRMFrame::AddChildƒƒƒƒ,,,　,,,,‘,ƒƒ　ƒ,’‰
,,,,,,,,　Žƒƒ　ƒ,Š‘,　　,,,,IDirect3DRMFrame::De*lete*Childƒƒƒƒ,Ž—
,,　Žƒƒ　ƒ,　ƒƒ　ƒ,Ž“,,,,　IDirect3DRMFrame::GetChildren,IDirect3DR*MF*
*ra*me::GetParentƒƒƒƒ,Ž—,,

ƒƒ□ƒ,□‘,ƒƒ□ƒ,ƒƒƒƒƒ□ƒƒƒƒƒƒ,,,’‰,,　—,,,,Š‘　‘,　*ƒ□ƒ*‘‘,“,,*‰*“,——
,,,,,*‰*”,,,　　,,Š‘,*ƒƒƒƒƒƒ*,Œ,,,□*ƒƒƒ□ƒƒƒ*,’‰,–
,,,　　*ƒƒ□ƒ*,Ž*ƒƒ□ƒ*,“,　,,,,,,’ˆ,•—
,,,□•Ž*ƒ□ƒ*,□Ž□*Ž,*□*‰Š*‘,*ƒƒƒƒ,*□*,,,*□□‰Š‘,□□,　IDirect3DRMFrame の

のの

4　4 の[x, y, z, 1]のの

vchild

vparent=vchildTchild

Tchild

IDirect3DRMFrame::AddTransform
Direct3DRMFrame::AddScale　IDirect3DRMFrame::AddRotation
IDirect3DRMFrame::AddTranslation の　　　　　　　　　　　　　　　　　　　の
D3DRMCOMBINETYPE　　のの　　　　　　　　　　　　　　　　　　　のの

IDirect3DRMFrame::**GetRotation**　**IDirect3DRMFrame::GetTransform** の
の　　　　　　　　**IDirect3DRMFrame::SetRotation**

IDirect3DRMFrame::Transform
IDirect3DRMFrame::InverseTransform

のDirect3D
の3D

0 □□□□□□□

の

□*f*,‘,

IDirect3DRMFrame::AddMoveCallback

Direct3DRMFrame::DeleteMoveCallback

のの
ののの の

IDirect3DRMLight IDirec*t3*DRMLightArray

,, ,,,,ffffff, ,, ffffff, , ff
ffff',ŠŒŒ,‰‹, —,,

Œ’ の

ののののの

のXの
の *Direct3D*
の

5の

IDirect3DRMLight
IDirect3DRMLight

Direct3D

-
-
-
-

•

のの
 ののの

の

の

の

の

ののののの
 の
IDirect3DRMLight::GetPenumbra *IDirect*3DRMLight::GetUmbra
IDirect3DRMLight::SetPenumbra IDirect3DRMLight::SetUmbra

§

IDirect3DRMMaterial

ののemissive specular□,,,,,□□—‘,,, ” ののののの
 5 のの

の
IDirect3DRMMaterial::GetEmissive IDirect3DRMMaterial::SetEmissive
IDirect3DRMMaterial::GetSpecular
IDirect3DRMMaterial::SetSpecular
IDirect3DRMMaterial::GetPower IDirect3DRMMaterial::SetPower

IDirect3DRMMaterial ののIDirect3DRMMaterial

IDirect3DRMMesh IDirect3**DRMMeshBuilder**

のの
のののの
のの
ののの のの2D

IDirect3DRMMesh IDirect3DRMMeshBuilde*r* のの COM
IDirect3DRMMesh

IDirect3DRMMeshBuilder IDirect3DRMMesh の
IDirect3DRMMeshBuilder
Direct3DRMMeshBuilder
Direct3DRMMesh
の
のの

の
IDirect3DRMMesh

のの
IDirect3DRMMesh::AddGroup の
ののの

IDirect3DRMMeshBuilder IDirect3DRMMesh の
64 ののののののDirect3DRMMes**h**
Direct3DRMMeshBuilder API Direct3D API

IDirect3DRMMeshBuilder::AddVertex
IDirect3DRMMeshBuilder::AddFace
IDirect3DRMMeshBuilder::AddFaces

IDirect3DRMMesh::SetGroupColor
IDirect3DR**MMesh::SetGroupColorRGB**
IDirect3DRMMesh::SetGroupTexture IDirect3DRMMesh::SetGroupMaterial
ののののの 1

IDirect3DRMFrame::AddVisual

IDirect3DRMMesh::SetGroupQuality ののの
D3DRMRENDERQUALITY の

のIDirect3DRMMeshBuilder::GenerateNormals

Direct3DRMObject

Direct3DRMObject ののDirect3DRMObject の

Direct3DRMObject **COM**
I**Unknown**

Direct3DRMCre**ate** **Direct3D** の
の
の**Direct3DRMAnimation**
IDirect**3DRM::CreateAnimation** の
のののの
ののの

のの 32 のの

IDirect3DRMObject::GetAppData のIDirect3DRMObject::SetAppData
Direct3DRMFrame の
のの
IDirect3DRMFrame::GetParent Direct3DRMFrame

の

ののの 1
IDirect3DRMObject::SetName
IDirect3DRMObject::GetName

のののののののの

の

IDirect3DRMObjec*t::AddDe*stroyCallback の
IDirect3DRMObject::DeleteDestroyCallback

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□ **0** □□□□□□□□□□□□□□□□□の□□□□□□□□□□□□□□□□□□□□□□
のの
□□,”,,,,,,,,□
Š˜□•„„„□IDirect3DRMObject

IDirect3DRMPickedArray

2D
IDirect3DRMPickedArray
IDirect3DRMViewport::Pick
IDirect3DRMPickedArray::GetPick
IDirect3DRMFrameArray
の
ののの
の

IDirect3DRMShadow

IDirect3DRM::C**reateShadow** の
IDirect3DRMShadow の
IDirect3DRM::CreateObject
IDirect3DRMShadow::Init

IDirect3DRMTexture

ののののの

IDirect3DRMTexture **DirectDrawSurface**
Direct3D のDirect3D の DirectDraw の
のDirect3D の

D3DRMIMAGE
IDirect3DRM::CreateTexture
IDirect3DRM::CreateTextureFromSurface DirectDraw
IDirect3DRM::LoadTexture

Windows の .bmp
.ppm

IDirect3DRMWrap Interface

の **IDirect3DRMTexture**

Direct3D の

-
- の
-
-
- の

の

IDirect3DRMTexture::SetDecalSize IDirect3DRMTexture::SetDecalSize の

の IDirect3DRMTexture::SetDecalOrigin
IDirect3DRMTexture::GetDecalOrigi**n** ののの**[0, 0]**

IDirect3DRMTexture::SetColors IDirect3DRMTexture::GetColors

RGB 8 24 32の

8のの

IDirect3DRMTexture::SetShades
IDirect3DRMTexture::GetShades

Direct3DRMTexture
D3DRMIMAGE
D3DRMIMAGE

IDirect3DRMDevice::SetTextureQuality

の

のののののの

IDirect3DRMDevice::SetTextureQuality
D3DRMTEXTUREQUALITY

の

□□□□□□□□□□□□□□□□□□□ IDirect3DRMTexture::SetDecalTransparency のDirectDraw □□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□□□□□□□□□□□□□ □□□□□□□ □□□□□□□□□□□□□ □□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□,,,Ž’,,,,,,,,□

DirectDraw,ƒƒ□□ƒ□□ƒƒ□ƒ,,,,□,,,□□ƒƒ□□ƒ□ƒƒƒ□,Ž□,,

IDirect3DRMTexture

IDirect3DRMUserVisual

の

IDirect3DRM::**CreateUserVisual**
IDirect3DRMUserVisual::Init
の

IDirect3DRMViewport and IDirect3DRMViewportArray

3D の 2D
の

のの の IDirect3DRMViewport

のの の

-
- Ž□‘
-
-

Dirct3DRMFrame
の
z **y** の

IDirect3DRMViewport::SetCamera のののの
IDirect3DRMViewport::GetCamera

‚3D*ƒƒƒ*□*ƒ*„□Ž□‘‚’„„*ƒƒƒƒƒƒ*‚‰Ž□‘„„□“Ž“‰„□*ƒƒƒƒ*□*ƒƒƒƒƒƒ*□のの の

§

のの z D の
F ののIDirect3DRMViewport::SetF**ront**
IDirect3DRMViewport::SetBack **ID**irect3DRMViewport::GetFront
IDirect3DRMViewport::GetBack の 2h の h の
IDirect3DRMViewport::SetField IDirect3DRMViewport::GetField

の A の
の h の

§

D3DRMPROJECTIONTYPE
IDirect3DRMViewport::GetProjection
IDirect3DRMView**port::SetProjection**

3D 2D
4 のの[x y z w]
[x y z w] 3 の[x/w y/w **z/w]** **[x/w y/w]**
z/w 0 1 の0
1 ののの

ののh F の z D の z

§

Direct3D 4,4 の

のののs のo の

§

の の P
W の

§

sx sy ox oy [-h -h D] [h h D]のの

ƒ •,*ƒƒƒƒ,*•Š,,,, IDirect3DRMViewport::Transform,IDirect3DRMViewport::Inver*seTrans*form*ƒƒƒƒ,*Ž—,, Ž,—,,,, *ƒƒƒƒ ƒƒƒ,* ,,,,*ƒƒƒƒ,*——
,,*ƒƒƒƒ,ƒƒ ƒ*,,,,,,,,

```
/*
 * Drag a frame by [delta_x delta_y] pixels in the view.
 */
void DragFrame(LPDIRECT3DRMVIEWPORT view,
   LPDIRECT3DRMFRAME frame,
   LPDIRECT3DRMFRAME scene,
   int delta_x, int delta_y)
{
   D3DVECTOR p1;
   D3DRMVECTOR4D p2;

   frame->GetPosition(scene, &p1);
   view->Transform(&p2, &p1);
   p2.x += delta_x * p2.w;
```

```
    p2.y += delta_y * p2.w;
    view->InverseTransform(&p1, &p2);
    frame->SetPosition(scene, p1.x, p1.y, p1.z);
}
```

ののの
*IDi*rect3DRMViewport::Transform の 4 の
の

の 3D のの**[x y z w]**の
§

Direct3D
3D

の**2D** の
IDirect3DRMViewport::Pick の

IDirect3DRMVisual and IDirect3DRMVisualArray

IDirect3DRMFrame::AddVisual

の IDirect3DRMVisualArray
IDirect3DRMVisu*al COM*

ƒƒƒƒƒƒ‚ƒƒƒ‚□Direct3DRMMeshBuilder
Direct3DRMTexture

IDirect3DRMWrap

の

IDirect3DRM::CreateWrap
IDirect3DRMWrap の
IDirect3DRMWrap::Apply IDirect3DRMWrap::ApplyRelative のの
IDirect3DRMWrap::Apply のIDirect3DRMWrap::ApplyRelative
の

v z u y
[0, 0, 0]

IDirect3DRMWrap
IDirect3DRMWrap

4

•

•

•

•

•

D3DRMMAPPIN**G** **D3DRMMAP_WRAPU**
D3DRMMAP_WRAPV の
,ƒƒƒƒƒ□•Š,□'‹—
, u v の
のの u
v のののの

- ƒƒƒƒƒ ƒƒƒ,ƒƒƒ,,,,,,•–ƒƒƒƒƒ ƒ ƒ,, u,,,v •,Ž',,,•–,, ƒƒƒƒƒ,–
  Œ,ƒƒƒƒƒ,,, ,, —Œ,u,,,v •',1.0ˆ ,,, "(0.1, 0.1),(0.9, 0,9)
  ,Œ, ',' , "(0.5, 0.5),'‰,,
- D3DRENDERSTATE_WRAPU,,,D3DRENDERSTATE_WRAPV,,,,,,ƒƒƒ,
  ,, ƒƒƒƒƒ,‰Ž1.0, ,,,,,‰"Œ,,, 1.0ˆ ,ƒƒƒƒƒ •, ƒƒƒ,,,,,,"ˆ",,—
  Œ,,, ƒƒƒƒƒ •Š, '‹—
  ,ƒƒƒƒƒ ƒƒƒ,,,,ˆ,, D3DRENDERSTATE_WRAPU,ƒƒƒ,,,,,,,, "(0.1, 0.1)
  ,,(0.9, 0.9),,, ',' , " 0, 0.5 ,'‰,,
- D3DRENDERSTATE_WRAPU,D3DRENDERSTATE_WRAPVƒƒƒ,—
  •,ƒƒƒ,,,,,, ƒƒƒƒƒ,‰Š' ƒ ƒƒ ,,, ƒƒƒƒ,• ,,,,,, 1.0ˆ ,ƒƒƒƒƒ •,–
  Œ,,, "(0.1, 0.1),,(0.9, 0.9),,, ',' , "(0, 0),'‰,,

–Œ—ˆ,,,,ƒƒƒƒƒ •,—Œ—ˆ,Ž,,,,,, ,,,,," ,• ,,,,,,

ˆ",ƒƒƒƒ ƒƒƒ, ƒƒƒƒƒ,Œ ,–,•,ˆ',,,,, ‰' ,ƒƒƒ,,,,,ƒƒƒƒƒ ƒƒƒ,ƒƒƒ,, ,,,,–
,ƒƒƒƒƒ,"•ˆ ,Ž—,,, ,, ƒƒƒƒƒ ƒƒƒ, ',,,

•–

•–ƒƒƒ,, ƒƒƒƒƒ, ƒƒƒƒƒƒ ,ƒƒƒƒƒ,,,ƒƒ,,,,ƒƒƒƒƒƒ,–,ˆ',,

Ž,"Ž,,, ƒƒƒƒ[x y z],,[u v] •,‹,,,,

u=sux–ou

v=svy–ov

,,,,ŒŽ,,,, s,*ƒƒƒƒƒ*,*ƒƒ ƒƒƒ*Œ o,*ƒƒƒƒƒ*,Œ“,Ž,,,, *ƒƒƒƒ ƒƒƒ*, 1‘,*ƒƒ ƒƒƒ* Œ , u,,,v,—Œ,’,Ž,,,,x,y,0,,1,”ˆ,*ƒƒƒ*,,,,,*ƒƒƒƒƒ*,Œ’,,,,,,,,,

‰’

‰’*ƒƒƒ*,, *ƒƒƒƒƒ*,‰’,‰,,•,Ž,,,,ˆ,, ,, ’,‰’,,,,,,,, *ƒƒƒƒƒƒ*,‰Ž,’‰ ,”’,, *ƒƒƒƒƒƒ*,*ƒ ƒƒƒ*, ,,,*ƒƒƒƒƒ*,•Œ,,,

‰ , ‰’*ƒƒƒƒƒ ƒƒƒ*,,,,Š*ƒƒƒƒ*,Œ‰,Ž,,,,,,,

§

•Œ*ƒƒƒƒ*,‰’,Ž,Ž,□□*ƒƒƒƒ*,‰“,Š‘,u = 0,,,“,Ž,□*ƒƒƒƒ*,□*ƒƒƒƒ*[x y z],‘,,*ƒƒƒƒƒ*,[u v]□•の

§

u v z 0 1 のv の

[x y 0] x のu v [x y z] z ののz

§

の

§

u v の 0 1 の

,”Ž,,Œ,,,,,*ƒƒƒƒƒ* •,Š,“,, *ƒƒƒ* *ƒƒƒ*,Ž *ƒƒ ƒ*,ˆ’,Ž“,

”Ž の のの の のu v

Direct3D のの

W*in*dows の Direct3D の の Windows □„,,, のののの の

§

のの

- Helworld.c

-

- Windows の

  - の
  - 3D
  -
  - の

-

Helworld.c

3D
の Direct3D のの
のの のの .c
DirectX
SDK の Sphere3.x
Hello.ppm の *ƒƒƒƒ*,—ˆ, 3D
の
の Direct3D の
SDK の

の Direct3D のの

-

-

  - DirectDraw の

の

のHelworld.c Helworld.c Helworld.c DirectX
SDK の Shp*e*r*e*3.*x* Hello.ppm の

のDirectX SDK の Globe の —‰
„„SDK の Direct3D のGlobe Rmmain.cpp Helworld.c
Rmmain.cpp の C++ C

ののののののの"Hello, world!" 3D の

ƒƒƒƒƒƒƒƒ

Direct3D,•Žƒƒƒƒƒƒƒƒƒƒ,ƒƒƒƒƒƒƒƒƒƒ Winmn.lib D3drm.lib

DirectDraw の

の Direct3D DirectDraw
のDirectDraw のの

のHelworld.c Helwold.c ののの

INITGUID の define DirectX

```
////////////////////////////////////////////////////////////////////
//
//  Copyright (C) 1996 Microsoft Corporation. All Rights Reserved.
//
//  File: Helworld.c
//
//  "Globe" SDKƒƒƒƒ,Š,,,   Š',Direct3D•Žƒ   ƒ,ƒƒƒƒ
//
////////////////////////////////////////////////////////////////////

#define INITGUID              // ',ƒƒƒ'‹,ƒƒƒƒ   ƒ,',
                              // '‹,,,,,,,,,
#include <windows.h>
#include <malloc.h>           // memset,Œ,   ,,•—
#include <d3drmwin.h>

#define MAX_DRIVERS 5         // D3Dƒƒƒƒ,   '
```

```
// ƒƒ   ƒƒ•

LPDIRECT3DRM lpD3DRM;          // Direct3DRMƒƒƒƒƒƒ
LPDIRECTDRAWCLIPPER lpDDClipper;// DirectDrawClipperƒƒƒƒƒƒ

struct _myglobs {
   LPDIRECT3DRMDEVICE dev;     // Direct3DRMƒƒƒƒ
   LPDIRECT3DRMVIEWPORT view;  // ƒ   ƒ‚•Ž‚‚Direct3DRMƒƒ   ƒ   ƒ
   LPDIRECT3DRMFRAME scene;    //
‘‚ƒƒƒƒƒƒ‚"'„„ƒƒƒ□□ƒƒ□ƒ
   LPDIRECT3DRMFRAME camera;   // の POV


   GUID DriverGUID[MAX_DRIVERS];     //           D3D の GUID

   char DriverName[MAX_DRIVERS][50]; //           D3D の

   int  NumDrivers;                 //           D3D の

   int  CurrDriver;                //                         D3D の


   BOOL bQuit;                 //

   BOOL bInitialized;          // の D3DRM

   BOOL bMinimized;            //


   int BPP;                    // のの


} myglobs;


// の


static BOOL InitApp(HINSTANCE, int);

long FAR PASCAL WindowProc(HWND, UINT, WPARAM, LPARAM);

static BOOL EnumDrivers(HWND win);

static HRESULT WINAPI enumDeviceFunc(LPGUID lpGuid,

   LPSTR lpDeviceDescription, LPSTR lpDeviceName,
   LPD3DDEVICEDESC lpHWDesc, LPD3DDEVICEDESC lpHELDesc,

   LPVOID lpContext);

static DWORD BPPToDDBD(int bpp);

static BOOL CreateDevAndView(LPDIRECTDRAWCLIPPER lpDDClipper,
   int driver, int width, int height);
static BOOL SetRenderState(void);
static BOOL RenderLoop(void);
static BOOL MyScene(LPDIRECT3DRMDEVICE dev,
    LPDIRECT3DRMVIEWPORT view,
   LPDIRECT3DRMFRAME scene, LPDIRECT3DRMFRAME camera);
void MakeMyFrames(LPDIRECT3DRMFRAME lpScene,
LPDIRECT3DRMFRAME lpCamera,

   LPDIRECT3DRMFRAME * lplpLightFrame1,

   LPDIRECT3DRMFRAME * lplpWorld_frame);

void MakeMyLights(LPDIRECT3DRMFRAME lpScene,
LPDIRECT3DRMFRAME lpCamera,

   LPDIRECT3DRMFRAME lpLightFrame1,

   LPDIRECT3DRMLIGHT * lplpLight1, LPDIRECT3DRMLIGHT *
lplpLight2);

void SetMyPositions(LPDIRECT3DRMFRAME lpScene,

   LPDIRECT3DRMFRAME lpCamera, LPDIRECT3DRMFRAME
lpLightFrame1,

   LPDIRECT3DRMFRAME lpWorld_frame);

void MakeMyMesh(LPDIRECT3DRMMESHBUILDER * lplpSphere3_builder);

void MakeMyWrap(LPDIRECT3DRMMESHBUILDER sphere3_builder,

         LPDIRECT3DRMWRAP * lpWrap);

void AddMyTexture(LPDIRECT3DRMMESHBUILDER lpSphere3_builder,

          LPDIRECT3DRMTEXTURE * lplpTex);

static void CleanUp(void);
```

Windows の

のHelworld.c   Windows の

- WinMain

- InitApp
-

WinMain

Helworld.c の WinMain DirectDraw Direct3D のInitApp CleanUp Windows の Helworld.c の Direct3D WinMain RenderLoop のRenderLoop ののRenderLoop

```
////////////////////////////////////////////////////////////////////////
//
// WinMain
//                                                ¥
//
//
////////////////////////////////////////////////////////////////////////


int PASCAL
WinMain (HINSTANCE this_inst, HINSTANCE prev_inst, LPSTR cmdline,
   int cmdshow)
{
   MSG     msg;
   HACCEL  accel = NULL;
   int     failcount = 0;  // RenderLoop


   prev_inst;
   cmdline;


   // の
   //


   if (!InitApp(this_inst, cmdshow))
```

```
            return 1;

        while (!myglobs.bQuit) {

            //

            while (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE)) {
                if (!TranslateAccelerator(msg.hwnd,
        accel, &msg)) {
                    TranslateMessage(&msg);
                    DispatchMessage(&msg);
                }
            }

            //
            // D3DRM

            if (!myglobs.bMinimized && !myglobs.bQuit &&
                myglobs.bInitialized) {

                // の 2
                //

                if (!RenderLoop())
                        ++failcount;
                    if (failcount > 2) {
                        CleanUp();
                        break;
                    }
                }
            }
            return msg.wParam;
        }

InitAppŠ
```

Helwold.c,□Š‰Š□,□,,,,,Windowsƒƒƒƒ□ƒƒƒ,‟—
,□ƒƒƒƒƒ□ƒƒƒ,‟˜,□ƒƒƒ□ƒƒƒƒ□ƒƒƒ□ƒƒƒƒƒ,□□,,□,,Œ□DirectDraw,Direct3D,—
,,ƒƒƒƒ□ƒƒƒ,‟—,□—,□,□
InitAppŠ□,□,,□Œ□,ƒƒƒƒƒƒ,ƒƒƒƒ‟,,,ƒƒƒ□,Ž‟,,□,,’,□ƒƒƒƒ□ƒƒƒ,ƒƒƒƒƒƒ,•Ž,□’,
,□,Ž—,,,□□,,,□ƒƒƒƒƒƒ□ƒƒ□ƒ,□’□,Ž□,,,,□
,,Œ□,,Direct3Dƒƒƒƒ,—
Œ,,,,,Œ’,□‟□,ƒƒƒƒ,‘‘,,,,,□ƒ□ƒƒ’‹,EnumDriversŠ□,Œ,□,□ƒƒƒƒ,—‹,,,,,□,,,□□
ƒƒƒƒ□ƒƒƒƒ,—‹□,Ž□,,,,□
Ž,□Direct3DRMCreateŠ□,Œ,□,□IDirect3DRM
ƒƒƒ□ƒƒƒƒ,□□,,□,,ƒƒƒ□ƒƒƒƒ,□ƒ□ƒ,ƒƒƒ□ƒƒ□ƒ,□□,,,□ƒƒƒ,ƒ□ƒ,□’,,,,□IDirect3D
RM::CreateFrame,IDirect3DRMFrame::SetPosition,Œ,□,,,,Ž—,,,□
DirectDrawClipperƒƒƒƒƒƒ,□3Dƒ□ƒ,‰Ž••,□Œ,,ƒƒƒƒƒƒ ƒƒ□ƒ,Š—
,Š’,,,□Helworld.c,,□IDirectDrawClipperƒƒƒ□ƒƒƒƒ,□□,,,,,DirectDrawCreat
eClipperŠ□,Œ,□,□IDirectDrawClipper::SetHWndƒƒƒƒ,Ž—
,,□ƒƒƒƒƒƒ□•,Ž‟,,ƒƒƒƒƒ,ƒƒƒƒ,□’,,□
,,,□ƒ□ƒƒ’‹,CreateDevAndViewŠ□,Œ,□,□Direct3Dƒƒƒƒ,ƒƒ□ƒ□ƒ,□□,,□,,Š□,,,,□
,,,□ƒƒƒƒ,ƒƒ□ƒ□ƒ,□□□,Ž□,,,,□
Direct3Dƒƒƒƒ□ƒƒƒ,ƒƒ□ƒ,,,,,,□‘‘,□Š‰,Š—,,,□3Dƒ□ƒ,□,,□□,,,,,,,,,,,,,□,,□
—,MySceneŠ□,□,,,,□MySceneŠ□,,,,□,,,□ƒƒƒƒ,ƒƒ□ƒ□ƒ,□□□,Ž□,,,,□
□Œ,InitAppŠ□,□•□‟,□Š‰Š□,‟—,□ƒƒƒƒƒ,•Ž,□□,□,□

```
//////////////////////////////////////////////////////////////////////
//
// InitApp
// ƒƒƒƒƒ,□□,□ƒƒƒƒƒƒ,ŠŽ,,,,,•—,,,,,ƒƒƒƒƒƒ,
// □Š‰,,□
//
//////////////////////////////////////////////////////////////////////

static BOOL
InitApp(HINSTANCE this_inst, int cmdshow)
{
    HWND win;
    HDC hdc;
    WNDCLASS wc;
    RECT rc;

    // ƒƒƒƒƒƒ,□,□ƒƒƒƒƒ□ƒƒƒ,‟˜,,□

    wc.style = CS_HREDRAW | CS_VREDRAW;
    wc.lpfnWndProc = WindowProc;
    wc.cbClsExtra = 0;
    wc.cbWndExtra = sizeof(DWORD);
    wc.hInstance = this_inst;
    wc.hIcon = LoadIcon(this_inst, "AppIcon");
    wc.hCursor = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground = (HBRUSH)GetStockObject(BLACK_BRUSH);
    wc.lpszMenuName = NULL;
    wc.lpszClassName = "D3DRM Example";
    if (!RegisterClass(&wc))
        return FALSE;

    // ƒƒ□ƒƒ•□,□Š‰,,□

    memset(&myglobs, 0, sizeof(myglobs));
    // ƒƒƒƒƒ‚□□‚‚□

    win =
        CreateWindow
        (   "D3DRM Example",            // ƒƒƒƒƒ□ƒƒƒ
            "Hello World (Direct3DRM)", // ƒƒƒƒƒ□
            WS_VISIBLE | WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU |
                 WS_MINIMIZEBOX | WS_MAXIMIZEBOX,
            CW_USEDEFAULT,              // □Šx□•
            CW_USEDEFAULT,              // □Šy□•
            300,                        // □Š‚•
            300,                        // □Š‚□‚
            NULL,                       // □ƒƒƒƒƒ
            NULL,                       // ƒƒƒ□□ƒƒƒƒ
            this_inst,                  // ƒƒƒƒƒ‚ƒƒƒƒƒƒ□ƒƒƒƒ
            NULL                        // □□ƒƒƒ□ƒ
        );
    if (!win)
        return FALSE;

    // Œ□‚ƒƒƒƒƒƒ‚ƒƒƒƒ"‚‚‚ƒƒƒ□‚‹‰‚‚□

    hdc = GetDC(win);
    myglobs.BPP = GetDeviceCaps(hdc, BITSPIXEL);
    ReleaseDC(win, hdc);

    // D3Dƒƒƒƒ‚¬‹‚□‚‚‚‚``‚‚□

    if (!EnumDrivers(win))
        return FALSE;

    // D3DRMƒƒƒƒƒƒ‚D3DRMƒƒƒƒƒƒ‚□□‚‚□

    lpD3DRM = NULL;
    Direct3DRMCreate(&lpD3DRM);

    // ƒƒƒ□□ƒ□ƒ‚ƒƒ□ƒ‚ƒƒƒ□ƒƒ□ƒ‚□□‚‚□

    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, NULL, &myglobs.scene);
    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, myglobs.scene,
        &myglobs.camera);
    myglobs.camera->lpVtbl->SetPosition(myglobs.camera,
myglobs.scene,
        D3DVAL(0.0), D3DVAL(0.0), D3DVAL(0.0));

    // DirectDrawClipperƒƒƒƒƒƒ‚□□‚□ƒƒƒƒƒ‚Š˜•‚‚□

    DirectDrawCreateClipper(0, &lpDDClipper, NULL);
    lpDDClipper->lpVtbl->SetHWnd(lpDDClipper, 0, win);

    // ``‚‚‚D3Dƒƒƒƒ‚–‚‚D3DRMƒƒƒƒ‚□□‚‚□

    GetClientRect(win, &rc);
    if (!CreateDevAndView(lpDDClipper, myglobs.CurrDriver, rc.right,
            rc.bottom)) {
        return FALSE;
    }

    // ƒƒƒƒƒƒ,,,ƒ□ƒ,□□,,□

    if (!MyScene(myglobs.dev, myglobs.view, myglobs.scene,
            myglobs.camera))
        return FALSE;

    myglobs.bInitialized = TRUE;  // □Š‰Š—

    // ƒƒƒƒƒ,•Ž,,□

    ShowWindow(win, cmdshow);
    UpdateWindow(win);

    return TRUE;
}
```

ƒƒƒ□ƒƒƒƒƒ□ƒƒƒ□ƒƒ
Helworld.cƒƒƒƒ,ƒƒƒ□ƒƒƒƒƒ□ƒƒƒ□ƒƒ,”□,’□,,,□Ž□,□,,ƒƒƒƒ,ƒ□ƒ,“—
,‵,Ž,•,,,ƒƒƒƒ□ƒƒƒ,,,□
ƒƒƒƒƒ□ƒƒƒ□ƒƒ,□WM_DESTROYƒƒƒ□ƒ,Ž,Ž,,CleanUpŠ□,Œ,□,□
,,WM_ACTIVATEƒƒƒ□ƒ,Ž,Ž,,,,,,□ƒƒƒƒƒ□ƒƒƒ□ƒƒ,IDirect3DRMWinDevice,Ž“,□ƒƒ
ƒƒƒ,ƒƒƒƒƒƒ□ƒƒƒƒƒ,□,□,,•Ž,,,,,IDirect3DRMWinDevice::HandleActivateƒƒƒƒ
,Œ,□,,□,□“—
,□WM_PAINTƒƒƒ□ƒ,‰“,,□ƒƒƒƒƒ□ƒƒƒ□ƒƒ,IDirect3DRMWinDevice::HandlePaintƒƒ
ƒƒ,Œ,□,□

```
//////////////////////////////////////////////////////////////////////
//
// WindowProc
// ƒƒƒ□ƒƒƒƒƒ,ƒƒƒ□ƒ□ƒƒƒƒ
//
//////////////////////////////////////////////////////////////////////

LONG FAR PASCAL WindowProc(HWND win, UINT msg,
    WPARAM wparam, LPARAM lparam)
{
    RECT r;
    PAINTSTRUCT ps;
    LPDIRECT3DRMWINDEVICE lpD3DRMWinDev;

    switch (msg)    {

    case WM_DESTROY:
        CleanUp();
        break;

    case WM_ACTIVATE:
        {

        // ,,ƒƒƒ□ƒ,□—,,□ƒƒƒƒƒŒ—,D3DRMƒƒƒƒƒ□ƒƒƒƒ,
        // □□,,□

        LPDIRECT3DRMWINDEVICE lpD3DRMWinDev;
        if (!myglobs.dev)
            break;
        myglobs.dev->lpVtbl->QueryInterface(myglobs.dev,
            &IID_IDirect3DRMWinDevice, (void **) &lpD3DRMWinDev);
        lpD3DRMWinDev->lpVtbl->HandleActivate(lpD3DRMWinDev,
            (WORD) wparam);
        lpD3DRMWinDev->lpVtbl->Release(lpD3DRMWinDev);
        }
        break;

    case WM_PAINT:
        if (!myglobs.bInitialized || !myglobs.dev)
            return DefWindowProc(win, msg, wparam, lparam);

        // ,,ƒƒƒ□ƒ,□–,,□ƒƒƒƒƒŒ–,D3DRMƒƒƒƒƒ□ƒƒƒƒ,
        // □□,,□

    if (GetUpdateRect(win, &r, FALSE)) {
        BeginPaint(win, &ps);
        myglobs.dev->lpVtbl->QueryInterface(myglobs.dev,
         &IID_IDirect3DRMWinDevice, (void **) &lpD3DRMWinDev);
      if (FAILED(lpD3DRMWinDev->lpVtbl->HandlePaint(lpD3DRMWinDev,
         ps.hdc)))
      lpD3DRMWinDev->lpVtbl->Release(lpD3DRMWinDev);
      EndPaint(win, &ps);
    }
    break;
  default:
    return DefWindowProc(win, msg, wparam, lparam);
  }
  return 0L;
}
```

の

Direct3D のの

- EnumDrivers

- enumDeviceFunc
- BPPToDDBD

EnumDrivers

EnumDrivers　　　　InitApp の

IDirect3D COM □□□□□□□□□□□□□□□ DirectDraw □□□□□□□□□□□□□□□□□□□□□□□□□□ の DirectDrawCreate □□□□□□□□ DirectDraw □□□□□□□□□□□□□□□□ EnumDrivers □□□□ QueryInterface □□□□□□□□ IDirect3D □□□□　　　　　　*C*　　*Query*Interface □□□□□ □□□□□□□□‘2ƒƒƒ□ƒ,,□□C++
,,ƒƒƒƒƒƒƒ,,,,□’□,,,,,’,Ž’,,,,,,,□ƒƒƒ□ƒƒƒƒ,Ž•Ž,ƒƒƒƒ,“,,,,,,,,,,□
ƒƒƒƒ,—‹,□IDirect3D::EnumDevicesƒƒƒƒ,,,,□,,,□IDirect3D::EnumDevicesƒƒƒ
ƒ,□ƒ□ƒƒ,’‹,,,enumDeviceFuncƒ□ƒƒƒƒŠ□,——
,,□,,ƒ□ƒƒƒƒŠ□,,,,□,,,□□enumDeviceFuncƒ□ƒƒƒƒŠ□□,Ž□,,,,□
IDirect3D::EnumDevices,Direct3Dƒƒƒƒ,,,□Direct3DRMƒƒƒƒ,,,,,,,,’ˆ,•—
,,,□•Žƒ□ƒ,API,,□—‹,□,ƒƒƒƒ,‘□,,,□,,,□,,,,ƒƒƒƒ□ƒƒƒ,•Žƒ□ƒ,’□ƒ□ƒ,—•,Ž—
,,□□,—,—,,,□

```
//////////////////////////////////////////////////////////////////////
//
// EnumDrivers
// —Œ,D3Dƒƒƒƒ,—‹,□,,,,‘‘,,□
//
//////////////////////////////////////////////////////////////////////

static BOOL
EnumDrivers(HWND win)
{
    LPDIRECTDRAW lpDD;
    LPDIRECT3D lpD3D;
    HRESULT rval;

    // DirectDrawƒƒƒƒƒƒ,□□,□ƒƒƒƒ,—‹,—,,Direct3D
    // ƒƒƒ□ƒƒƒƒ,–,□,,,□

    DirectDrawCreate(NULL, &lpDD, NULL);
    rval = lpDD->lpVtbl->QueryInterface(lpDD, &IID_IDirect3D,
        (void**) &lpD3D);
    if (rval != DD_OK) {
        lpDD->lpVtbl->Release(lpDD);
        return FALSE;
    }

    // enumDeviceFunc,ƒƒƒƒ‘‘ƒ□ƒ,□Š‰,,,,□CurrDriver,
    // -1,□’,□ƒƒƒƒ,—‹,,□

    myglobs.CurrDriver = -1;
    lpD3D->lpVtbl->EnumDevices(lpD3D, enumDeviceFunc,
        &myglobs.CurrDriver);

    // □,,,,—Œ,ƒƒƒƒ,,,,,,,,,,,•□,,□
    if (myglobs.NumDrivers == 0) {
        return FALSE;
    }
    lpD3D->lpVtbl->Release(lpD3D);
    lpDD->lpVtbl->Release(lpDD);

    return TRUE;
}
```

enumDeviceFuncƒ□ƒƒƒƒŠ□
enumDeviceFunkŠ□,□D3DENUMDEVICESCALLBACKŒ,ƒ□ƒƒƒƒŠ□,,,□D3DENUMDEVICESC
ALLBACKŒ,ƒƒƒ□ƒƒƒƒD3dcaps.h,'‹,,,,,□ƒƒƒƒ,,,Š□,□ƒƒƒƒ□ƒ,,,,,ŠDirect3Dƒƒƒ
ƒ,Ž•Ž,–'□,,,ƒ□ƒƒƒƒ,ƒƒƒƒ□ƒ,,,ƒƒƒƒ,"—,'',,□
ƒ□ƒƒƒƒŠ□,□D3DDEVICEDESC□'',dcmColorModelƒƒƒ,Ž—
,,□ƒ□ƒƒƒƒ,—‹,,,ƒƒƒƒ,,,,,',,,,Œ',,□,,ƒƒƒ,ƒ□ƒƒƒƒ,□',,,,,□□□Š□,ƒ□ƒƒƒƒ,□"
,',,□
Ž,□ƒ□ƒƒƒƒŠ□,□—‹,,,ƒƒƒƒ,Œ□,ƒƒ□
□ƒƒƒ□,ƒƒƒƒƒƒ,□,,,,,,,,,,,,"',,□•‰",□□,D3DENUMRET_OK,•,□,,ƒƒƒƒ,Š,,Ž,,□
—,ƒƒƒƒ,,□Ž,ƒƒƒƒ,—‹,',,□ƒ□ƒƒƒƒŠ□,□ƒ□ƒƒ'‹,BPPToDDBDŠ□,—,□'',,,ƒƒ□
□ƒƒƒ□,□InitAppŠ□,,GetDeviceCapsŠ□,Œ,□,,,,,Ž",,,□‰'",,"Š,,□BPPToDDBD,b
its-per-pixel to DirectDraw bit-depth,—
,,,□□BPPToDDBDŠ□,ƒ□ƒ,,,,,□□BPPToDDBDƒƒƒŠ□□,Ž□,,,,□
—‹,,,ƒƒƒƒ,,,,Š',ƒƒƒ,□,,Œ,□D3DDEVICEDESC□'',',ƒƒƒ,"',,□ƒ□ƒƒƒƒŠ□,□ƒƒƒƒƒ
ƒ ƒƒƒƒ□ƒƒƒ,,,ƒ□ƒƒƒƒ,□ƒƒƒƒ□ƒƒƒ,,,RGBƒƒƒ,'',,□

```
////////////////////////////////////////////////////////////////////////
//
// enumDeviceFunc
// Ž—‰",D3Dƒƒƒƒ,–',GUID,‹‰,,ƒ□ƒƒƒƒŠ□□
// ƒƒƒƒ,'',□*lpContext,□',,□
//
////////////////////////////////////////////////////////////////////////

static HRESULT
WINAPI enumDeviceFunc(LPGUID lpGuid, LPSTR lpDeviceDescription,
    LPSTR lpDeviceName, LPD3DDEVICEDESC lpHWDesc,
    LPD3DDEVICEDESC lpHELDesc, LPVOID lpContext)
{
    static BOOL hardware = FALSE; // Œ□,ŠŽƒƒƒƒ,ƒ□ƒƒƒƒ,,,
    static BOOL mono = FALSE;     // Œ□,ŠŽƒƒƒƒ,ƒƒƒƒŒŒ,,,
    LPD3DDEVICEDESC lpDesc;
    int *lpStartDriver = (int *)lpContext;

      // ,,ƒƒƒƒ‹□,',,,,Œ'„□

  lpDesc = lpHWDesc->dcmColorModel ? lpHWDesc : lpHELDesc;

  // ⓓⓓⓓ
  //
```

```
    if (!(lpDesc->dwDeviceRenderBitDepth & BPPToDDBD(myglobs.BPP)))
        return D3DENUMRET_OK;

    // のの GUID

    memcpy(&myglobs.DriverGUID[myglobs.NumDrivers], lpGuid,
        sizeof(GUID));
    lstrcpy(&myglobs.DriverName[myglobs.NumDrivers][0], lpDeviceName);

    //                                                   RGB

    if (*lpStartDriver == -1) {

        // の

        *lpStartDriver = myglobs.NumDrivers;
        hardware = lpDesc == lpHWDesc ? TRUE : FALSE;
        mono = lpDesc->dcmColorModel & D3DCOLOR_MONO ? TRUE :
FALSE;
    } else if (lpDesc == lpHWDesc && !hardware) {

        // の

        *lpStartDriver = myglobs.NumDrivers;
        hardware = lpDesc == lpHWDesc ? TRUE : FALSE;
        mono = lpDesc->dcmColorModel & D3DCOLOR_MONO ? TRUE :
FALSE;
    } else if ((lpDesc == lpHWDesc && hardware ) ||
            (lpDesc == lpHELDesc && !hardware)) {
        if (lpDesc->dcmColorModel == D3DCOLOR_MONO && !mono) {
```

// の **RGB**

//

***lpStartDriver = myglobs.NumDrivers;**

hardware = lpDesc == lpHWDesc ? TRUE : FALSE;

mono = lpDesc->dcmColorModel & D3DCOLOR_MONO ? TRUE : FALSE;

}

}

myglobs.NumDrivers++;

if (myglobs.NumDrivers == MAX_DRIVERS)

return (D3*DENUMRET_CANCEL);*

return (D3DENUMRET_OK);

}

BPPToDDBD

enumDeviceFunc BPPToDDBD
の
enumDeviceFunc enumDeviceFunc

//

//

// BPPToDDBD

// の DirectDraw の

//

//

```
static DWORD
BPPToDDBD(int bpp)
{
    switch(bpp) {
        case 1:
            return DDBD_1;
        case 2:
            return DDBD_2;
        case 4:
            return DDBD_4;
        case 8:
            return DDBD_8;
        case 16:
            return DDBD_16;
        case 24:
            return DDBD_24;
        case 32:
            return DDBD_32;
        default:
            return 0;
    }
}
```

3DŠ‹,ƒƒƒƒƒƒ

,,ƒƒƒƒƒ,,□Helworld.c,ƒ□ƒ,,,□3DŠ‹,□’,,••,,,,□–,,□ˆ‰,ƒƒƒƒƒ,□,,□—,Ž□,,,,,,,,,Š□,,,,□–,,,,□

- ƒƒƒƒ,ƒƒ□ƒ□ƒ,□□
- ƒƒƒƒƒƒ□ƒƒ□ƒ,□’

„„Š□,□3DŠ‹,ƒƒƒƒƒƒ,ƒƒ□ƒ□Œ,"’„„„„□ƒ□ƒ,□□,□MySceneŠ□,□MySceneŠ□,Œ,□,Š□Œ„„ŽŒ„ □3DŠ‹,ƒ□ƒ,□’„•–„„„□□ƒ□ƒ,□□□,Ž□„„□

ƒƒƒƒ,ƒƒ□ƒ□ƒ,□□

Direct3Dƒƒƒƒ,ƒƒ□ƒ□ƒ,□ƒƒƒƒ□ƒƒƒ,□Š‰

,ˆŠ„□□„□InitAppŠ□,□DirectDrawClipperƒƒƒƒƒƒ,□□„Œ□DirectDrawClipperƒƒƒƒƒƒ,“„ƒƒƒƒ□, „ƒƒƒƒƒƒ‹Œ,□–,ƒƒƒ□ƒ„□CreateDevAndViewŠ□,Œ,□,□

CreateDevAndViewŠ□,□—‹ƒƒƒƒ,“„ƒƒƒƒ,Ž— „□IDirect3DRM::CreateDeviceFromClipperƒƒƒƒ„„Direct3DRMƒƒƒƒ,□□„□„IDirect3DRMDeviceƒƒƒ□ƒƒƒƒ,□ƒƒƒƒ,•„□„Ž“„IDirect3DRMDevice::GetWidth,IDirect3DRMDevice::GetHeightƒƒƒ ƒ,Œ,□„Ž„□CreateDevAndViewŠ□,□ƒƒƒƒ,•,□„□•,Ž,Ž„Œ□IDirect3DRM::CreateViewportƒƒƒƒ, Œ,□„IDirect3DRMViewportƒƒƒ□ƒƒƒƒ,Ž“„□

Ž,CreateDevAndViewŠ□,IDirect3DRMViewport::SetBackƒƒƒƒ„„ƒƒ□ƒ□ƒ,ƒƒƒ□ƒƒƒƒƒƒ□ƒƒ□ƒ, □’„□ƒ□ƒƒ’‹,SetRenderStateŠ□,Œ,□„□SetRenderStateŠ□„„□Ž,ƒƒƒƒƒ□ƒƒƒƒƒƒ□ƒƒ□ƒ,□’□, □–„□

```
////////////////////////////////////////////////////////////////////////
//
// CreateDevAndView
// Ž’„D3Dƒƒƒƒ„ƒƒƒ,D3DRMƒƒƒƒ,ƒƒ□ƒ□ƒ,□□„□
//
////////////////////////////////////////////////////////////////////////

static BOOL
CreateDevAndView(LPDIRECTDRAWCLIPPER lpDDClipper, int driver,
    int width, int height)
{
    HRESULT rval;

    // Ž’„D3Dƒƒƒƒ,—,□„ƒƒƒƒƒ„D3DRMƒƒƒƒ,□□„□

    lpD3DRM->lpVtbl->CreateDeviceFromClipper(lpD3DRM, lpDDClipper,
        &myglobs.DriverGUID[driver], width, height, &myglobs.dev);

    // ƒƒƒ□ƒƒ□ƒ,Ž„D3DRMƒƒ□ƒ□ƒ,□□„□”Œ,□„‘„□,
    // □’„□•,□„„„’□„„„„□ƒƒƒƒ„Ž“„□
    width = myglobs.dev->lpVtbl->GetWidth(myglobs.dev);
    height = myglobs.dev->lpVtbl->GetHeight(myglobs.dev);
    rval = lpD3DRM->lpVtbl->CreateViewport(lpD3DRM, myglobs.dev,
        myglobs.camera, 0, 0, width, height, &myglobs.view);
    if (rval != D3DRM_OK) {
        myglobs.dev->lpVtbl->Release(myglobs.dev);
        return FALSE;
    }
    rval = myglobs.view->lpVtbl->SetBack(myglobs.view, D3DVAL(5000.0));
    if (rval != D3DRM_OK) {
        myglobs.dev->lpVtbl->Release(myglobs.dev);
        myglobs.view->lpVtbl->Release(myglobs.view);
        return FALSE;
    }

    // ƒƒƒƒƒƒ‚•Ž□“„„„ƒ□ƒ□ŒŒ‚□‘□ƒƒ□□ƒƒ□ƒƒƒƒ‚□•‚
    // □’„□

    if (!SetRenderState())
        return FALSE;
    return TRUE;
}
```

ƒƒƒƒƒƒ□ƒƒ□ƒ‚□’
Direct3D ののの
□
SetRenderState

SetRenderState
IDirect3DRMDevice::SetQuality
ののの
IDirect3DRMDevice::SetDither
IDirect3DRMDevice::SetTextureQuality

の のの **switch** のの**IDirect**3DRMDevice::SetShades
IDirect3DRM::SetDefaultTextureColors
IDirect3DRM::SetDefaultTextureShades

```
/////////////////////////////////////////////////////////////////////

//

// SetRenderState

// の

//

/////////////////////////////////////////////////////////////////////


BOOL
```

```
SetRenderState(void)
{
   HRESULT rval;

   //                ꩺ

   rval = myglobs.dev->lpVtbl->SetQuality(myglobs.dev,
      D3DRMLIGHT_ON | D3DRMFILL_SOLID |
D3DRMSHADE_GOURAUD);
   if (rval != D3DRM_OK) {
      return FALSE;
   }

   // □□□□□□□□□□□□□□□□□□□□□□ SetDither □□□□,□

    // ƒƒƒƒƒ,•Ž,D3DRMTEXTURE_NEAREST□ƒƒƒƒƒ□ˆŠ,,,,□□□
    // ,,,SetTextureQuality,Œ,□,□

    // Œ□,ƒƒƒƒ‟,,,ƒƒƒ□,Š,,,□‰‰ƒ□ƒ,□’,,□

    switch (myglobs.BPP) {
        case 1:
            if (FAILED(myglobs.dev->lpVtbl->SetShades(myglobs.dev,
4)))
                goto shades_error;
            if (FAILED(lpD3DRM->lpVtbl->
                    SetDefaultTextureShades(lpD3DRM, 4)))
                goto shades_error;
            break;
        case 16:
            if (FAILED(myglobs.dev->lpVtbl->SetShades(myglobs.dev,
32)))
                goto shades_error;
            if (FAILED(lpD3DRM->lpVtbl->
                    SetDefaultTextureColors(lpD3DRM, 64)))
                goto shades_error;
            if (FAILED(lpD3DRM->lpVtbl->
                    SetDefaultTextureShades(lpD3DRM, 32)))
                goto shades_error;
            break;
        case 24:
        case 32:
            if (FAILED(myglobs.dev->lpVtbl->
                    SetShades(myglobs.dev, 256)))
                goto shades_error;
            if (FAILED(lpD3DRM->lpVtbl->
                    SetDefaultTextureColors(lpD3DRM, 64)))
                goto shades_error;
            if (FAILED(lpD3DRM->lpVtbl->
                    SetDefaultTextureShades(lpD3DRM, 256)))
                goto shades_error;
            break;
    }
    return TRUE;
shades_error:
    return FALSE;
}
```

ƒƒƒƒƒƒ□ƒ□ƒ
WinMainŠ□,□Ž,ƒƒ□ƒ,•‰,,,,□RenderLoopŠ□,Œ,□,□RenderLoopŠ□,□,,,,,’□,□—
,Ž□,,□

- IDirect3DRMFrame::Moveƒƒƒƒ,Œ,□,□Š‘‰,,,,,ƒƒ□ƒ,‰“,“‘,“—,□,□
- Direct3DRMViewport::Clearƒƒƒƒ,Œ,□,□Œ□,ƒƒ□ƒ□ƒ,”Œ□,ƒƒƒ,,□
- IDirect3DRMViewport::Renderƒƒƒƒ,Œ,□,□Œ□,ƒ□ƒ,ƒƒ□ƒ□ƒ,ƒƒƒƒƒƒ,,□
- IDirect3DRMDevice::Updateƒƒƒƒ,Œ,□,□ƒƒƒƒƒƒ,,,ƒƒ□ƒ,ƒƒƒ□ƒ,ƒƒ□,,□

```
///////////////////////////////////////////////////////////////////////
//
// RenderLoop
// ƒƒ□ƒ□ƒ,ƒƒƒ,,Ž,ƒƒ□ƒ,ƒƒƒƒƒƒ,□ƒƒƒƒƒ,□□,,□
//
///////////////////////////////////////////////////////////////////////

static BOOL
RenderLoop()
{
    HRESULT rval;

    // Œ□,ƒ□ƒ,Š”,,□

    rval = myglobs.scene->lpVtbl->Move(myglobs.scene, D3DVAL(1.0));
    if (rval != D3DRM_OK) {
        return FALSE;
    }

    // ƒƒ□ƒ□ƒ,ƒƒƒ,,□

    rval = myglobs.view->lpVtbl->Clear(myglobs.view);
    if (rval != D3DRM_OK) {
        return FALSE;
    }

    // ƒ□ƒ,ƒƒ□ƒ□ƒ,ƒƒƒƒƒƒ,,□

    rval = myglobs.view->lpVtbl->Render(myglobs.view, myglobs.scene);
    if (rval != D3DRM_OK) {
        return FALSE;
    }
    // ƒƒƒƒƒ‚„‚

    rval = myglobs.dev->lpVtbl->Update(myglobs.dev);
    if (rval != D3DRM_OK) {
        return FALSE;
    }
    return TRUE;
}
```

ƒƒ‚

3DŠ‹‚ƒƒƒƒƒƒƒƒƒ‚“3Dƒƒƒ‚ƒƒƒƒ‚ƒƒƒƒƒƒƒƒ‚’„‚Š—„„Helworld.c‚„3DŠ‹‚ƒƒƒƒƒƒ‚ƒƒƒŒ‚”„„„ŠŒ‚Œ‚‚

- MySceneŠ
- MakeMyFramesŠ
- MakeMyLightsŠ
- SetMyPositionsŠ
- MakeMyMeshŠ
- MakeMyWrapŠ
- AddMyTextureŠ

MySceneŠ

Helworld.c‚MySceneŠ‚DirectX SDK„„„Direct3Dƒƒƒ‚ƒƒƒƒƒƒƒ„„„BuildSceneŠ‚“„ƒƒƒƒƒƒƒ‚ƒƒƒƒƒƒ‚ƒƒƒƒƒ‚–Œ‰„„•Ž„—‚„„„Š“‚„„

MySceneŠ‚„„ƒƒ‚Š“‚’„ƒƒƒ’‹‚ŠŒ‚Œ‚‚‚„„Š‚ˆ‰‚Ž‚

- MakeMyFrames
- MakeMyLights
- SetMyPositions
- MakeMyMesh
- MakeMyWrap
- AddMyTexture

‚‚‚Š‚ƒƒƒƒƒƒƒƒƒƒƒ‚ƒƒƒƒƒƒ‚Š—‚‚‚MySceneŠ‚IDirect3DRMFrame::AddVisualƒƒƒ‚Œ‚‚ƒƒƒƒƒƒ‚3DŠ‹，world ƒƒƒ‚’‰„‚„Œ‚‚‚ Release の

```
////////////////////////////////////////////////////////////////////
//
// MyScene
// ƒƒ  ƒ  ŒŒ  ƒƒƒƒ  ƒƒƒƒƒ‚    ‚‚Š  ‚Œ‚  ‚  Š—‚‚‚  ‚‚‚‚
// ƒƒƒ  ƒƒƒƒ‚‰•‚‚
//
////////////////////////////////////////////////////////////////////

BOOL
MyScene(LPDIRECT3DRMDEVICE dev, LPDIRECT3DRMVIEWPORT view,
    LPDIRECT3DRMFRAME lpScene, LPDIRECT3DRMFRAME lpCamera)
```

```
{
    LPDIRECT3DRMFRAME lpLightframe1 = NULL;
    LPDIRECT3DRMFRAME lpWorld_frame = NULL;
    LPDIRECT3DRMLIGHT lpLight1      = NULL;
    LPDIRECT3DRMLIGHT lpLight2      = NULL;
    LPDIRECT3DRMTEXTURE lpTex       = NULL;
    LPDIRECT3DRMWRAP lpWrap         = NULL;
    LPDIRECT3DRMMESHBUILDER lpSphere3_builder = NULL;

    MakeMyFrames(lpScene, lpCamera, &lpLightframe1, &lpWorld_frame);
    MakeMyLights(lpScene, lpCamera, lpLightframe1, &lpLight1,
      &lpLight2);
    SetMyPositions(lpScene, lpCamera, lpLightframe1, lpWorld_frame);
    MakeMyMesh(&lpSphere3_builder);
    MakeMyWrap(lpSphere3_builder, &lpWrap);
    AddMyTexture(lpSphere3_builder, &lpTex);

    // ƒƒƒƒƒ‚□□‚‚•—‚‚‚□□□‚‚‚‚□Œ‚‹‚–‚□□‚‚□□□□
    // ‚‚‚CreateMaterial‚SetMaterial‚Œ‚□‚□

    // ‚‚‚ƒƒƒƒƒ□ƒƒƒƒƒƒ‚□□‚‚‚‚‚□ƒ□ƒƒ□ƒƒ□ƒ‚’‰‚‚□

    lpWorld_frame->lpVtbl->AddVisual(lpWorld_frame,
        (LPDIRECT3DRMVISUAL) lpSphere3_builder);

    lpLightframe1->lpVtbl->Release(lpLightframe1);
    lpWorld_frame->lpVtbl->Release(lpWorld_frame);
    lpSphere3_builder->lpVtbl->Release(lpSphere3_builder);
    lpLight1->lpVtbl->Release(lpLight1);
    lpLight2->lpVtbl->Release(lpLight2);
    lpTex->lpVtbl->Release(lpTex);
    lpWrap->lpVtbl->Release(lpWrap);

    return TRUE;
}
```

MakeMyFramesŠ□
MySceneŠ□‚MakeMyFramesŠ□‚Œ‚□‚□Helworld.c‚—
‚‚‚‚ƒƒƒƒƒƒƒƒŒŒƒƒ□ƒ‚ƒ□ƒƒ□ƒƒ□ƒ‚□□‚‚□MakeMyFramesŠ□‚□IDirect3DRM::Create
Frameƒƒƒƒ‚Œ‚□‚‚□‚‚□—‚Ž□‚‚□

```
////////////////////////////////////////////////////////////////////////
//
// MakeMyFrames
// ƒ□ƒ‚Ž—‚‚ƒƒ□ƒ‚□□‚‚□
//
////////////////////////////////////////////////////////////////////////

void MakeMyFrames(LPDIRECT3DRMFRAME lpScene, LPDIRECT3DRMFRAME
lpCamera,
    LPDIRECT3DRMFRAME * lplpLightFrame1,
    LPDIRECT3DRMFRAME * lplpWorld_frame)
{
    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, lpScene, lplpLightFrame1);
    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, lpScene, lplpWorld_frame);
}
```

MakeMyLightsŠ
MySceneŠ,MakeMyLightsŠ,Œ,,Helworld.c,—
,,,,ƒƒƒƒƒƒƒŒŒ,ƒƒƒƒƒƒƒƒƒ,,,,MakeMyLightsŠ,IDirect3DRM::CreateLightRGB,IDirect3DRMFrame::AddLightƒƒƒƒ,Œ,,•Œ,Ž,,–,,ŒŒ,,,,,,ƒƒƒ,'‰
,,,,"ˆ,ƒƒƒƒƒƒƒƒƒ,,ƒƒ``,'‰,,ƒƒƒƒƒƒƒƒƒ,,,,ƒƒ``,Š˜•,,,,

```
////////////////////////////////////////////////////////////////////
//
// MakeMyLights
```

// ƒƒ,Ž—,,ŒŒ,,„

//

//

```
void MakeMyLights(LPDIRECT3DRMFRAME lpScene,
LPDIRECT3DRMFRAME lpCamera,
    LPDIRECT3DRMFRAME lpLightFrame1,
    LPDIRECT3DRMLIGHT * lplpLight1, LPDIRECT3DRMLIGHT *
      lplpLight2)
{
    lpD3DRM->lpVtbl->CreateLightRGB(lpD3DRM,
      D3DRMLIGHT_DIRECTIONAL,
        D3DVAL(0.9), D3DVAL(0.9), D3DVAL(0.9), lplpLight1);

    lpLightFrame1->lpVtbl->AddLight(lpLightFrame1, *lplpLight1);

    lpD3DRM->lpVtbl->CreateLightRGB(lpD3DRM,
      D3DRMLIGHT_AMBIENT,
        D3DVAL(0.1), D3DVAL(0.1), D3DVAL(0.1), lplpLight2);

    lpScene->lpVtbl->AddLight(lpScene, *lplpLight2);
}
```

SetMyPositionsŠ

MySceneŠ ,SetMyPositionsŠ ,Œ, , Helworld.c,Ž,ƒƒ ƒ,ˆ',Œ,, ',, SetMyPositionsŠ , ,, —
,IDirect3DRMFrame::SetPosition,,,IDirect3DRMFrame::SetOrientationƒƒƒƒ

,Œ, ,,,,,Ž ,, IDirect3DRMFrame::SetRotation*ƒƒƒƒ*, ‹‘,’‰ ,,,*ƒƒ ƒ*,‰“, ’,,

```
////////////////////////////////////////////////////////////////////////

//

// SetMyPositions

// の

//

//

////////////////////////////////////////////////////////////////////////


void SetMyPositions(LPDIRECT3DRMFRAME lpScene,

   LPDIRECT3DRMFRAME lpCamera, LPDIRECT3DRMFRAME
lpLightFrame1,

   LPDIRECT3DRMFRAME lpWorld_frame)

{


   lpLightFrame1->lpVtbl->SetPosition(lpLightFrame1, lpScene,

      D3DVAL(2), D3DVAL(0.0), D3DVAL(22));

   lpCamera->lpVtbl->SetPosition(lpCamera, lpScene,
      D3DVAL(0.0), D3DVAL(0.0), D3DVAL(0.0));
   lpCamera->lpVtbl->SetOrientation(lpCamera, lpScene,
      D3DVAL(0.0), D3DVAL(0.0), D3DVAL(1),
      D3DVAL(0.0), D3DVAL(1), D3DVAL(0.0));

   lpWorld_frame->lpVtbl->SetPosition(lpWorld_frame, lpScene,
      D3DVAL(0.0), D3DVAL(0.0), D3DVAL(15));
   lpWorld_frame->lpVtbl->SetOrientation(lpWorld_frame, lpScene,
      D3DVAL(0.0), D3DVAL(0.0), D3DVAL(1),
      D3DVAL(0.0), D3DVAL(1), D3DVAL(0.0));


   lpWorld_frame->lpVtbl->SetRotation(lpWorld_frame, lpScene,

      D3DVAL(0.0), D3DVAL(0.1), D3DVAL(0.0), D3DVAL(0.05));

}
```

MakeMyMesh

MyScene □□□□Helworld.c □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ MakeMyMesh □□□□ □□□□MekeMyMesh □□□ IDirect3DRM::C**reateMeshBuild**er □□□□□□□□□□□□ IDirect3DRMMeshBuilder □□□□□□□□□□□□□□□□□□□□ IDirect3DRMMeshBuilder::Load□IDirect3DRMMeshBuilder::Scale□‚‚‚IDirect3DRMMeshBuilder::SetColorRGBƒƒƒƒ‚Œ‚□‚□Sphere3.xƒƒƒƒ‚ Ž‚ƒƒƒƒ‚—ˆ‚‚□Sphere3.xƒƒƒƒ‚□ƒƒƒƒ□ƒ□ƒ‚Ž—‚‚ƒƒƒƒ‚‚‚‚‚‚DirectX SDK‚Ž˜‚‚‚‚‚□□

```
////////////////////////////////////////////////////////////////////
//
// MakeMyMesh
// MeshBuilderƒƒƒƒƒƒ‚□□‚□ƒ□ƒ□ƒƒ□ƒƒƒ□ƒƒƒƒ‚□•‚‚□‚□
//
////////////////////////////////////////////////////////////////////

void MakeMyMesh(LPDIRECT3DRMMESHBUILDER * lplpSphere3_builder)
{
    lpD3DRM->lpVtbl->CreateMeshBuilder(lpD3DRM, lplpSphere3_builder);

    (*lplpSphere3_builder)->lpVtbl->Load(*lplpSphere3_builder,
        "sphere3.x", NULL, D3DRMLOAD_FROMFILE, NULL, NULL);

    (*lplpSphere3_builder)->lpVtbl->Scale(*lplpSphere3_builder,
        D3DVAL(2), D3DVAL(2), D3DVAL(2));

    // —'Š‚ƒƒƒƒƒ□ƒƒƒƒƒƒƒ‚”‚‚‚‚□‹'‚”‚□’‚‚□

    (*lplpSphere3_builder)->lpVtbl->SetColorRGB(*lplpSphere3_builder,
        D3DVAL(1), D3DVAL(1), D3DVAL(1));
}
```

MakeMyWrapŠ□

MySceneŠ□‚MakeMyWrapŠ□‚Œ‚□‚‚ƒƒƒƒƒ□•‚□□‚□MakeMyMeshŠ□‚ƒ□ƒ‚‚‹'‚"—‚‚□MakeMyWrapŠ□‚□‹'‚Š‚•Œ‹Š□‚Ž"‚‚‚‚‚IDirect3DRMMeshBuilder::GetBoxƒƒƒƒ‚Œ‚□‚□‚‚•Œ‹Š□‚ƒƒƒ‚IDirect3DRM::CreateWrapƒƒƒƒ‚Œ‚□‚Ž‚Ž—‚‚□IDirect3DRM::CreateWrapƒƒƒƒ‚‰’‚ƒƒƒƒƒ□ƒƒƒ‚□□‚□IDirect3DRMWrapƒƒƒ□ƒƒƒƒ‚Ž"‚‚□ƒƒƒƒƒ□•‚‹'‚"—‚‚‚‚□IDirect3DRMWrap::Applyƒƒƒƒ‚Œ‚□‚‚□‚□

```
////////////////////////////////////////////////////////////////////
//
// MakeMyWrap
// ƒƒƒ‚□□‚□ƒƒƒƒƒ‚"—‚‚□
//
////////////////////////////////////////////////////////////////////

void MakeMyWrap(LPDIRECT3DRMMESHBUILDER sphere3_builder,
                LPDIRECT3DRMWRAP * lpWrap)
{
    D3DVALUE miny, maxy, height;
    D3DRMBOX box;

    sphere3_builder->lpVtbl->GetBox(sphere3_builder, &box);

    maxy = box.max.y;
    miny = box.min.y;
    height = maxy - miny;

    lpD3DRM->lpVtbl->CreateWrap
        (lpD3DRM, D3DRMWRAP_CYLINDER, NULL,
        D3DVAL(0.0), D3DVAL(0.0), D3DVAL(0.0),
        D3DVAL(0.0), D3DVAL(1.0), D3DVAL(0.0),
        D3DVAL(0.0), D3DVAL(0.0), D3DVAL(1.0),
        D3DVAL(0.0), D3DDivide(miny, height),
        D3DVAL(1.0), D3DDivide(-D3DVAL(1.0), height),
        lpWrap);

    (*lpWrap)->lpVtbl->Apply(*lpWrap, (LPDIRECT3DRMOBJECT)
        sphere3_builder);

}

AddMyTextureŠ□
MySceneŠ□,AddMyTextureŠ□,Œ,□,,□ƒƒƒƒƒ,ƒ□ƒ,‹‘,,Š˜•,,□,□AddMyTextureŠ□,□
IDirect3DRM::LoadTextureƒƒƒƒ,—
,□Hello.ppm,,,–‘,ƒƒƒƒƒƒ,ƒ□ƒ,,□,,,,IDirect3DRMMeshBuilder::SetTexture,
Œ,□,□ƒƒƒƒƒƒ,‹‘,“,•,,□Hello.ppm,256□256,256□ƒƒƒƒƒƒ,,,□
     §
//////////////////////////////////////////////////////////////////////
//
// AddMyTexture
// ƒƒƒ,□□,□ƒƒƒƒƒ,“—,,□
//
//////////////////////////////////////////////////////////////////////

void AddMyTexture(LPDIRECT3DRMMESHBUILDER lpSphere3_builder,
    LPDIRECT3DRMTEXTURE * lplpTex)
{
    lpD3DRM->lpVtbl->LoadTexture(lpD3DRM, "hello.ppm", lplpTex);

    // ƒƒƒƒƒ□16□ˆŠ,ƒƒ□□“,•—,□□□,,,
    // IDirect3DRMTexture::SetShades,Œ,□,□

    lpSphere3_builder->lpVtbl->SetTexture(lpSphere3_builder,
*lplpTex);

}

□—□—
Helworld.c,□WM_DESTROYƒƒƒ□ƒ,Ž,Ž,,,,□,,,RenderLoopŠ□,Œ,□,,‰“,Ž”,,,,□Cl
eanUpŠ□,Œ,□,□
//////////////////////////////////////////////////////////////////////
//
// CleanUp
// ,,,,D3DRMƒƒƒƒƒƒ,‰•,□bQuitƒƒƒ,ƒƒƒ,,□
//
/////////////////////////////////////////////////////////////////////
```

```
void

CleanUp(void)

{

   myglobs.bInitialized = FALSE;

   myglobs.scene->lpVtbl->Release(myglobs.scene);

   myglobs.camera->lpVtbl->Release(myglobs.camera);

   myglobs.view->lpVtbl->Release(myglobs.view);

   myglobs.dev->lpVtbl->Release(myglobs.dev);
    lpD3DRM->lpVtbl->Release(lpD3DRM);
    lpDDClipper->lpVtbl->Release(lpDDClipper);

    myglobs.bQuit = TRUE;
}
```

’□ƒ□ƒ,Š–
’□ƒ□ƒ,,,,
,,ƒƒƒƒƒ,,□Microsoft,’ƒƒƒ3D API,,,’□ƒ□ƒ,,,,□–
,,□Direct3D,’□ƒ□ƒ,□ƒ□ƒ,,,‵,ƒƒ□ƒƒƒ□ƒƒƒ,ƒƒƒƒƒƒƒ□ƒ □□□□□□□
□□Microsoft Windows のの
□ Direct3D のの

の □□□□□□□□□□
の 3D の
□□□の□□□□□□□ 3D の
のDirect3D ののDirect3D のDirect3D
Direct3D
の : Direct3D のののの SDK

Direct3D の

のDirect3D のののの *Direct3D のの*

: Direc*t3D の*

Direct3D の □□□の□□□□□□□ののAPI
Direct3D の □□□□□□□□□の□ の
の Direct3D

のの8 の のCOM

COM

IDirect3D

COM

IDirect3DDevice

IDirect3DTexture

DirectDraw

IDirect3DMaterial

IDirect3DLight

IDirect3DViewport

IDirect3DDevice

4　4の

□□ƒƒƒƒ

IDirect3DExecuteBufferƒƒƒ□ƒƒƒƒ

’“ƒ□ƒ‚ƒƒƒ‚ƒƒƒƒƒƒ•–‚ŽŽ

ƒƒƒƒ

COM
のの
の

- *COM*
- COM ののの COM の
- の

,•, Direct3D,Š*ff*
***ffff*,‘ ,,,ŒŽ,Ž,**
,,,

fffff

COM*fff ffff*
‘‘

ffff

ffff

fffff

ŒŒ

ff f f

Texture*ffffff*

Material*ffffff*

—

Light*ffffff*

Ž *ffff*

Viewport*ffffff*

Direct3D*ffffff*Œ

,,fffff,, Direct3
D,ffffffŒ,fff
ffff,,,, –
,, ffff fff, Ž
,‘,,, , Direct3
D*ffffff,fff ff*
ff, ,,,,,,,,
Direct3Dfff fff
f ffffff

Deviceƒƒƒƒƒƒ

Execute-bufferƒƒƒƒƒƒ

IDirect3Dƒƒƒ ƒƒƒƒ

*IDirect*3D*ƒƒƒ ƒƒƒƒ,* DirectDraw*ƒƒƒƒƒƒ ,,ƒƒƒ ƒƒƒƒ,,,* IDirect3D*ƒƒƒ ƒƒƒƒ, ,,,, Ž,,,,IDirectDraw2::QueryInterfa*ce*ƒ ƒƒƒ,*Œ*, ,, ,*

lpDirectDraw->*QueryInterface(*

*IID_IDirect3D, // IDirect3Dƒƒƒ ƒƒƒƒ,*ID

lpD3D); // Direct3D*ƒƒƒƒƒƒ,ƒƒƒƒ*

IDirect3Dƒƒƒ ƒƒƒƒ,,,,Ž ,,,ƒƒƒƒƒƒ, ƒƒ ƒ ƒ ŒŒ *ƒƒƒƒƒ ,,,ƒƒƒƒ,ƒƒƒ ,*Š*,,,,* IDirect3D*,ƒƒƒƒ, ‘,ƒƒƒƒƒƒ, ,,, Direct3Dƒƒƒƒ,Œ,,,,,,Ž— ,,,,,,,,*

*IDirect*3DDevice*ƒƒƒ ƒƒƒƒ*

IDirect3DDevice*ƒƒƒ ƒƒƒƒ,* DirectDr aw

Surface の IDirect3DDevice の
IDirectDrawSurface2::QueryInterface のの IDirectDraw::CreateSurface
IDirectDrawSurface::GetAttachedSurface
IDirectDraw2 IDirectDrawSurface2 の QueryInterface

```
lpDirectDraw->CreateSurface(
   lpDDSurfDesc,   // DDSURFACEDESC の
   lpFrontBuffer,  // DIRECTDRAWSURFACE の
   pUnkOuter);     // NULL
lpFrontBuffer->GetAttachedSurface(
   &ddscaps,         // DDSCAPS  ‘‘,,ƒƒƒƒ
   &lpBackBuffer);  // DIRECTDRAWSURFACE   ‘‘,,ƒƒƒƒ
lpBackBuffer->QueryInterface(
   GUIDforID3DDevice,   // IDirect3DDeviceƒƒƒ  ƒƒƒƒ,ID
   lpD3DDevice);        // DIRECT3DDEVICEƒƒƒƒƒƒ,,ƒƒƒƒ
```

ƒƒƒƒƒƒƒ,,,,IDirectDrawSurface::QueryInterfaceƒƒƒƒ,Œ,□,,Ž’,,   ,ƒƒƒ□ƒ,□I Direct3DDeviceƒƒƒ□ƒƒƒƒ,Ž,□ƒƒ□ƒƒ□ƒƒƒ,— ˆ,Ž•Ž□GUID□,,,□,,GUID,□IDirect3D::EnumDevicesƒƒƒƒ,Œ,□,,,,,Ž“,,,,,,,, ƒƒƒ IDirect3D::EnumDevices のの D3DENUMDEVICESCALLBACK

ƒƒŠ ,Œ, ,,, ,,
GUID,‹,,

Direct3D,ƒƒƒƒ ƒ
ƒƒƒƒƒ, ƒƒƒ ƒƒƒ
ƒ ƒƒƒ,‘ , ,,,, ‘
,   Ž ,Ž ƒƒƒƒ
,ƒƒ ƒ ƒ,ƒƒƒ,Ž,,,
, ,, ƒƒƒƒƒ,ƒƒƒƒ
ƒ,ƒƒƒ,Š,,,, ,,,,ƒƒ
ƒ’,Ž,ƒƒƒƒƒ,ƒƒƒƒƒ,
,ƒƒƒƒ, ‘,ƒƒƒƒ,,ƒƒ
ƒƒ,—
•,•Ž,,,, ,,Š‘ ‘,,,,
,,, ƒƒƒƒƒƒ, ‘
,

Ž□,,,,□

IDirect3DDeviceƒƒƒ□ƒƒƒƒ,ƒƒ のののの

Direct3D の IDirect3DDevice::CreateMatrix IDirect3DDevice::SetMatrix

ƒƒƒƒ,Ž ƒƒƒƒ,—,,,,

IDirect3DTextureƒƒƒ ƒƒƒƒ

ƒƒƒƒƒ, •,,ƒƒƒƒ,‹Œ,•,,,,,,, ‹Œ,•,,, •Œ,,,, ,,,, ƒƒƒƒ, ,Œ
—“,“ ,,,, •Œ, ,,, ƒƒƒƒƒ,ƒƒƒƒƒ ƒƒƒƒ ƒ,–
*,,,Ž,,,,,,,, ,, , •Œ,,,,,,,,, R*GB*ƒƒ ƒƒ ƒ,—,,ƒƒƒƒ ƒƒƒ,, 8 24 32ƒƒƒ,ƒƒƒƒƒ,Ž— ,,,,,,,, ƒƒƒƒ ƒƒƒ ƒƒ ƒƒƒ,, 8ƒƒƒ,ƒƒƒƒ ƒ,,Ž*

,,

IDirect3DTexture*ƒƒƒ ƒƒƒƒ,* DirectDr*awSurface*
IDirect3DTexture
IDirectDrawSurface2::QueryInterface **IID_IDirect**3DTexture
Direct3D の DirectDraw の **Direct3D**

のIDirect3DTexture -

```
, IDirect3DTexture::GetHandle,,,IDirect3DTexture::Loadƒƒƒƒ,—
,,ƒƒƒƒƒ,ƒ ƒ,,•–,Ž,,,,
lpDDS->QueryInterface(IID_IDirect3DTexture,
    lpD3DTexture);  // DIRECT3DTEXTUREƒƒƒƒƒƒ,,ƒƒƒƒ
lpD3DTexture->GetHandle(
    lpD3DDevice,     // DIRECT3DDEVICE

  lphTexture);    // D3DTEXTUREHANDLE の

lpD3DTexture->Load(

  lpD3DTexture);  // DIRECT3DTEXTURE の
```

¥のののの

,ƒƒƒƒƒ,Ž ƒƒƒƒ,—,,, ƒƒ ƒ ƒ,,Zƒƒƒƒ,,,Ž,,, ƒƒƒƒƒ,ƒ のの
*ID*irect3DTexture

のDirect3D のの

•

•

•

•

のの

ののD**3DRENDERSTATETY**PE D3DRENDERSTATE_WRAPU D3DRENDERSTATE_WRAPV

の のの *u*

v の ののの

u **v** のの のの

- □□□ ƒƒ□ƒƒƒ,ƒƒƒ,,,,,,ƒƒƒƒ□ƒƒƒƒƒ□ƒ□ƒ,,□u,,,v□•,Ž',,,•–,,□ƒƒƒƒƒ,–Œ,ƒƒƒƒƒ,,,□,,□□□–Œ,u,,,v□•',1.0ˆ□,,,□"(0.1, 0.1),(0.9, 0,9),Œ,□',''□,□"(0.5, 0.5),'‰,,□
-

D3DRENDERSTATE_WRAPU,,,D3DRENDERSTATE_WRAPV,,,,,,ƒƒƒ,,,□□□ƒƒƒƒƒ,‰Ž1.0,□,,,,,‰"Œ,,,□1.0ˆ□,ƒƒƒƒƒ□•,□ƒƒƒ,,,,,,"ˆ",,—Œ,,,□ƒƒƒƒƒ□•Š,□'‹—,ƒƒƒƒƒ□ƒƒƒ,,,,ˆ,,□D3DRENDERSTATE_WRAPU,ƒƒƒ,,,,,,,,□"(0.1, 0.1),,(0.9, 0.9),,,□','□,□"□0, 0.5□,'‰,,□

- D3DRENDERSTATE_WRAPU,D3DRENDERSTATE_WRAPVƒƒƒ,—•,ƒƒƒ,,,,,,□ƒƒƒƒƒ,‰Š□□ƒ□ƒƒ□□,,,□ƒƒƒƒ,•,,,,,,□1.0ˆ□,ƒƒƒƒƒ□•,–Œ,,,□"(0.1, 0.1),,(0.9, 0.9),,,□','□,□"(0, 0),'‰,,□

–Œ—ˆ,,,,ƒƒƒƒƒ□•,—Œ—ˆ,Ž,,,,,,□,,,,,"□,•□,,,,,,□ˆ",ƒƒƒƒ□ƒƒƒ,□ƒƒƒƒƒ,Œ□,–,•,ˆ',,,,,□‰'□,ƒƒƒ,,,,,ƒƒƒƒƒ□ƒƒƒ,ƒƒƒ,,□,,,,–,ƒƒƒƒƒ,"•ˆ□,"—,,,□□,,□ƒƒƒƒƒ□ƒƒƒ,□',,,□
ƒƒƒƒƒ,,,,□,,,□□ƒƒƒƒƒƒƒƒƒ: Direct3D,•Žƒ□ƒ□ƒƒƒƒƒƒ□,□IDirect3DRMWrapƒƒƒ□ƒƒƒƒ□,Ž□,,,,□ƒƒƒ

のののの

D3DRENDERSTATETYPE
D3DRENDERSTATE_TEXTUREMAG
D3DRENDERSTATE_**TEXTUREMIN**

D3DRENDERSTATE_TEXTUREMAPBLEND
の

ののの
D3DTEXTUREBLEND

のD3DRENDERSTATE_SRCBLEND D3DRENDERSTATE_DESTBLEND

D3DBLEND

D3DTEXTUREFILTER の

の

D3DPRIMCAPS の **dwTextureFi**lterCaps

DirectDraw のの

D3DRENDERSTATETYPE の
D3DRENDERSTATE_BLENDENABLE

の **DirectDraw**

ののの

DirectDraw の

IDirect3DMaterial □□□□□□ ƒƒ

IDirect3DMaterialƒƒƒ□ƒƒƒƒ,□□„„□IDirect3D::CreateMaterial のの
IDirect3DMaterial のIDirect3DMaterial::SetMaterial
IDirect3DMaterial::GetHandle のの

lpDirect3D->CreateMaterial(

lplpDirect3DMaterial, //

pUnkOuter); // NULL

lpDirect3DMaterial->SetMaterial(

lpD3DMat); // D3DMATERIAL の

lpDirect3DMaterial->GetHandle(

lpD3DDevice, // DIRECT3DDEVICE の

lpD3DMat); // D3DMATERIAL の

ののの

の

のの

IDirect3DMaterial ののの

IDirect3DLight

IDirect3Dlight IDirect3D::CreateLight
の *IDirect3Dlight* の *I*Direct3DLight::SetLight

lpDirect3D->CreateLight(

lplpDirect3DLight, //

pUnkOuter); // NULL

lpDirect3DLight->S**etLight(**

lpLight); // D3DLIGHT の

```
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
IDirect3Dlightƒƒƒ□ƒƒƒƒ,Ž—,,□ŒŒ,Ž‘‘,□’,□,,,,,,,□
IDirect3DViewportƒƒƒ□ƒƒƒƒ
IDirect3DViewportƒƒƒ□ƒƒƒƒ,□IDirect3D::CreateViewportƒƒƒƒ,Œ,□,,□□,,□Ž,
—,□IDirect3DViewportƒƒƒ□ƒƒƒƒ,□□•–,Ž,,,,□,,□,,—
,□IDirect3DDevice::AddViewportƒƒƒƒ,,,ƒƒ□ƒ□ƒ,’‰
,,,,,□IDirect3DViewport::SetViewport□IDirect3DViewport::SetBackground
□IDirect3DViewport::AddLightƒƒƒƒ,—,,ƒƒ□ƒ□ƒ,ƒƒƒƒƒƒ,,•–,□–,,,,□
lpDirect3D->CreateViewport(
    lplpDirect3DViewport,  // □,,ƒƒ□ƒ□ƒ,,ƒƒƒƒ
    pUnkOuter);            // NULL
lpD3DDevice->AddViewport(
    lpD3DViewport)         // ƒƒ□ƒ□ƒ,ƒƒƒƒ,ƒƒƒƒ,,
lpD3DViewport->SetViewport(
     lpData);          // のの
```

// D3DVIEWPORT の

lpD3DViewport->SetBackground(

lphMat); // の D3DMATERIALHANDLE の

lpD3DViewpor*t->AddLight(*

*lpD*3DLight); // の

¥の

IDirect3DViewport

IDirect3DExecuteBuffer

ののŽ *ƒƒƒƒ,* –, Ž *ƒƒƒƒ* ,Ž ,,,,

IDirect3DExecuteBuffer*ƒƒƒ ƒƒƒƒ,* ,,,, IDirect*3DDevice::CreateExecut*eBuffer*ƒƒƒƒ*,Œ, ,, ,

lpD3DDevice->CreateExecuteBuffer(

lpDesc, *// DIRECT3DEXECUTEBUFFERDESC ‘‘,,ƒƒƒƒ*

*l*plpDirect3DExecuteBuffer, // Direct3DExecuteBuffer*ƒƒƒƒƒƒ*,,

// *ƒƒƒƒ*,Ž,Ž,,,,,*ƒƒƒƒ*

pUnkOuter); // NULL

Ž□ƒƒƒƒ‚ƒƒƒƒ□ƒƒƒ‚Š‚‚‚‚□Ž□‚ƒƒƒƒ—
ˆ‚Š•‚‚‚‚□IDirect3DDevice::CreateExecuteBuffer の

IDirect3DExecuteBuffer::Lock
IDirect3DExecuteBuffer::Unlock IDirect3DExecuteBuffer::SetExecuteData
の

lpD3DExBuf->Lock(

lpDes**c);.** // **DIRECT3DEXECUTEBUFFERDE**SC の

// **.**

// **. Store contents through the** su**pplied address**

// **.**

**lpD3DExBuf->Unlock(**);

lpD3DExBuf->SetExecuteData(

lpData); // D3DEXECUTEDATA の

のの IDirect3DE*xecu*teBuffer::SetExecuteData ののDirect3D
IDirect3DExecuteBuffer::Lock

IDirect3DExecuteBuffer

の

ƒƒ ƒ ƒ‚ —
‚‚‚‚ ‚‚‚‚‚‚ƒƒ ƒ ƒ‚‚‚‚‚‚ŠŒŒ‚•Ž‚‚‚‚‚‚Ž‚‚‚‚‚‚‚

§

ƒƒƒƒƒ‚ƒƒƒƒƒ‚ƒƒƒƒƒƒ‚ ‚‚‚ˆ ‚ƒƒƒƒ‚Š˜•‚‚‚‚‚‚‚‚ Ž‚ ‚Ž‚‚‚Šƒ ƒ‚ ƒƒƒ‚ “‚‚ƒƒƒƒ‚•
Ž‚‚‚‚ ‚‚‚ ‚’‚‚ƒƒƒƒ‚•Ž‚‚‚‚‚‚ ‚‚ƒƒƒƒƒ‚ƒ ƒ‚‚‚ ‚‚ƒƒƒƒƒ‚ƒ ƒ‚‚‚ ‚‚ƒƒƒƒƒƒ‚
“‚Š˜‚‚ƒƒƒƒ‚‚ƒƒƒƒ‚Ž“‚‚‚‚‚‚‚‚

§

ƒƒƒ ƒƒƒƒ‚ƒƒƒƒ ƒƒƒƒƒ‚ƒƒƒƒƒƒ‚ ‚‚‚‚QueryInterfaceƒƒƒƒ‚Œ‚ ‚ ƒƒƒƒƒ Œ
Œ ƒƒ ƒ ƒ‚ƒƒƒƒƒƒ‚ IDirect3Dƒƒƒ ƒƒƒƒ‚ƒƒƒƒ‚Œ‚ ‚‚ ‚‚‚‚‚‚‚‚ Ž ƒƒƒ
ƒ‚ —
‚ IDirect3DDeviceƒƒƒ ƒƒƒƒ‚‚‚‚ ‚‚‚ ƒƒƒ ƒƒƒƒ ƒƒƒƒƒƒ Direct3D
object ‚‚ ‚‚‚‚‚‚‚ƒƒƒƒ‚ƒƒ ƒ ƒ ŒŒ ƒƒƒƒƒ‚ƒƒƒƒƒƒ‚ƒƒƒ‚•Ž‚‚‚‚ Ž ƒƒ
ƒƒ‚ƒƒƒƒƒ‚ƒƒƒƒƒƒ‚Š‚‚‚‚

ƒ ƒ‚Š—

‚‚‚’ ƒ ƒ ƒƒƒƒ ƒƒƒ‚ ƒƒƒƒƒƒ‚Ž ‚‚‘‚IDirect3DDevice::BeginSceneƒƒƒƒ‚Œ‚
‚‚‚‚‚‚‚‚‚ ‚‚ ƒƒƒƒƒƒ‚Š—‚‚‚‚‚‚ IDirect3DDevice::EndSceneƒƒƒƒ‚Œ‚ ‚•—
‚‚‚ ƒƒƒƒ ƒƒƒ‚ Œ‚3Dƒ ƒƒƒƒ ‚“ ‚‚‚‚‚‚ ‚‚‚‚ƒƒƒƒ‚ ‚‚Ž—‚‚‚‚‚‚‚‚‚

ƒ ƒ,'",•‰*ƒƒƒƒƒƒ* ŽŠŒ ' " , ',,, '", "‰,*ƒƒƒ ƒƒƒ*,,,,,*ƒƒ ƒ*,•‰ ,,,,,Ž—
,,, IDirect3DDevice::BeginScene*ƒƒƒƒ*,*ƒ ƒ*,Ž,,, ,,,,,*ƒƒ ƒ*,ŠŽ, Ž, IDirect3DDevice::EndScene*ƒƒƒƒ*,*ƒ ƒ*,*ƒƒ ƒ*, —,'',, ,,,,*ƒƒ ƒ*,•‰ ,Ž,,,Ž *ƒƒƒƒ*——
, 1",IDirect3DDevice::BeginScene,IDirect3DDevice::EndScene,Œ, ,,Š,,,, ,,,,,,,

,,*ƒƒƒƒƒ*,ˆ‰,*ƒƒƒƒ*,,,, –,,,,

- ‰– ‹
- 2D,3D,'Œ —
- *ƒ ƒ*Š—*ƒƒƒ ƒƒ*

‰– ‹

3D*ƒƒƒƒƒ*□*ƒ*," *ƒƒƒ*,,□□—,Z*ƒƒƒƒ*
IDirect3DDevice::BeginScene
IDirect3DDevice::EndScene

のののの
の

ののののの
1 の *IDirect3DDevice::BeginScene IDirect3DDevi*ce::EndScene の
,,,,,,,,,□,,,,*ƒƒ*□*ƒ*,□□',□•□‰,IDirect3DDevice::BeginScene
IDirect3DDevice::EndScene の

のの
IDirect3DDevice::BeginScene IDirect3DDevice::EndScene

IDirect3DDevice::BeginScene **IDirect3DDevice::EndScene**

2D 3D の

IDirect3DDe**vice::BeginScene**
IDirect3DDevice::EndScene 3D のの 2D ののDirectDraw の
DDCAPS2_NO2DDURING3DSCENE のDirectDraw のGetDC の
IDirect3DDevice::EndScene の

のの 3D ののの

1 のの1 IDirect3DD**evice::BeginScene**
Direct3DDevice::EndScene

2 DirectDraw の DDCAPS2_NO2DDURING3DSCENE の□ □□□□□□□□□IDirect3DDevice::BeginScene の IDirect3DDevice::EndScene のの DirectDraw の *GetDC* □□□□□□□□□□□□の 2D □□□□ IDirect3DDevice::EndScene □□□□□□□□□□□□□□,,,,□

3 IDirect3DDevice::Execute*ƒƒƒƒ,* Œ,–
,,, ,,,,Ž *ƒƒƒƒ,*Š,,*ƒƒƒƒƒƒ ƒƒƒƒƒƒ,,,,,ƒ ƒƒƒ ƒ ƒƒƒ,ƒƒƒƒƒƒ,,,,*Œ,,,,,,,,
*ƒ ƒ ƒƒƒƒƒ ƒ ƒ, ƒ ƒ,*Š—
,,,, ,,, IDirect3DDevice::EndScene*ƒƒƒƒ,*Œ, ,,,,, *ƒƒƒƒƒƒ,*‰Š,,

Direct3D,' *ƒ ƒ,ƒƒ ƒƒƒƒ*

Windows*ƒ□ƒ,*Direct3D,'□*ƒ□ƒ□ƒƒƒƒ□ƒƒƒ,*□□,,,,□DirectDraw Direct3D の

SDK の

の SDK のの D3dmain.cpp D3dmain.cpp Windows のののののの Direct3D ののの

- Step 1:
- Step 2: DirectDraw Direct3D の
- Step 3: の
- Step 4: の
- Step 5: のの
- Step 6:
- Step 7: の
- Step 8:

の

SDK,',' *ƒ ƒ,ƒƒƒƒ,,* ,,,,‹ ,,,,,,*ƒ ƒ,,,,,*Š,,,,, "',,,SDK,, D3DappŠ ,,, Œ, ,,,*ƒƒƒŠ ,ƒƒƒƒƒƒ,*Š,,,,, "Ž,' *ƒ ƒ,ƒƒƒƒ ƒƒƒ,*‹ ,, ,—— ,,,,, ,,*ƒƒƒƒƒƒ,,ƒƒƒ*Š ,•",Ž ,,,,, ,,,,,,,,,*ƒƒ* ,,,,,,,,,, *ƒƒƒŠ , ƒ ƒ ƒƒ ƒƒ D3dapp.c Ddcalls.c D3dcalls.c Texture.c* ,,, *Misc.c ,ƒƒƒƒƒƒƒ,,,,, ƒ ƒ ƒƒƒƒ* Stats.cpp
, ƒƒ ƒ ƒ ƒ,ƒƒƒ ƒ ƒ ƒ, •,*ƒƒƒ ƒ,*',

D3main.cpp ,—,,,,,,*ƒƒƒƒ,* ,,"' *,ƒƒƒƒƒƒ,,,,,,,,,,* ˆ‰,*Š ,ƒƒƒƒƒƒƒ,,,,,,,,*

- InitScene
- *InitView*
- RenderScene
- Release*View*

- *ReleaseScene*
- OverrideDefaults

,,, *ƒƒƒƒ*,SetMouseCal*lbackŠ ,SetKeyboardCallbackŠ* ,Œ, , *ƒƒƒ,ƒ ƒ ƒ*,,, "—,Ž",,

Step 1: Š‰,ŠŽ

D3dmain.cpp,WinMainŠ□,□,□□,*ƒƒƒ,□ƒ□ƒƒ,'‹,,,AppInitŠ□,Œ,□,□ƒƒƒƒ□ƒƒ ƒ□ƒƒƒƒƒ,□□,□ƒƒƒƒƒƒ*,ŠŽ,•—,,,,,*ƒƒƒƒƒ*,□Š‰ ,,□,,□WinMainŠ□,D3dmain.cpp,*ƒƒƒ□ƒ□ƒƒƒ,ƒƒƒƒƒƒƒ,□ƒ□ƒƒ'‹,RenderLoopŠ □,C*leanUpAndPostQuit AppInit のののののの

Windows ののAppInit InitScene 3Dmain.cpp InitScene Oct1.c のInitScene TRUE Tunnel.c InitScene

AppInit D3dmain.cpp CreateD3DApp CreateD3DApp

Step 2: DirectDraw Direct3D の

D3dmain.cpp CreateD3DApp DirectDraw Direct3D CreateD3DApp のの D3DAppCreateFromHWND D3DAppGetRenderState OverrideDefaults D3DAppSetRenderState ReleaseView InitView の D3Dapp の D3Dapp のの ,,

WinMainŠ ,",,,*ƒƒƒƒ ƒƒƒ ƒƒƒƒƒ*,",,,, CreateD3DAppŠ ,,",,, — Œ,*ƒƒƒƒƒ*, -systemmemory,-emulation,,, -systemmemory*ƒƒƒƒƒ*, *ƒƒƒƒ* —,—,,,, -emulation*ƒƒƒƒƒ*,Ž',,, *ƒƒƒƒ ƒƒƒ*,DirectDraw,Direct3D,*ƒ ƒƒƒƒ ƒƒƒƒƒ ƒ*, Ž—,,,

CreateD3DAppŠ□,□*ƒƒƒƒƒ*,□□,,,,,D3DAppAddTextureŠ□,Œ,□,□,,,□D3DAppAddTextureŠ□,□*ƒ□ƒ□ƒƒƒƒƒ,ƒ□ƒƒƒ,ƒƒƒƒƒƒ,ƒƒƒƒ□ƒƒƒ*", ,, *ƒ□ƒƒƒƒ,ƒƒ ƒ□ƒƒƒ*,Ž,,,,□□,□□Š□',,‹,*ƒƒƒƒƒ□ƒ□ƒƒƒ,ƒƒƒ□ƒƒƒ*", ,, *ƒ□ƒ□ƒƒƒƒƒ*,□"' ,*ƒƒƒƒƒ□ƒ□ƒƒƒ,ƒ□ƒ*,,,Œ,"Š,,,□,,2'Š,*ƒƒƒƒ*,,,,□*ƒƒƒƒ*,□*ƒƒƒ□ƒƒƒ*,"',,,,,,*ƒƒƒƒ ƒ□ƒƒƒ*,ˆ□,,,□*ƒƒ□ƒƒƒ*,,,,,,,,□*ƒƒƒƒ□ƒ□ƒ*,,□IDirect3DTexture*ƒƒƒ□ƒƒƒƒ*,Ž",,, ,,IDirectDrawSurface::QueryInterface*ƒƒƒƒ*,Œ,□,□IDirect3DTexture::Load のIDirect3DTexture::GetHandle

のCreateD3DApp の DirectDraw Direct3D*ƒƒƒƒƒƒ*,□□,,□*ƒƒƒƒ□ƒ□ƒ*,,□D3DAppCreateFromHWND D3DAppCreateFromHWND D3dapp.c D3dcalls.c Texture.c Ddcalls.c

D3DAppCreateFromHWND DirectDrawEnumerate DirectDrawCreate DirectDraw のの IDirectDraw::EnumDisplayModes

IDirectDraw の IDirectDraw2 ののIDirectDraw IDirectDraw
IDirectDraw::EnumDisplayModes の
IDirectDraw2::EnumDisplayModes

D3DAppCreateFromHWND Direct3D Direct3D
Direct3D のIID_IDirect3D の
IDirectDraw::QueryInterface
IDirect3D::EnumDevices

IDirect3D::EnumDevices のDirect3D IDirect3D::FindDevice
のの
GUID
のの
の GUID

D3DAppCreateFromHWND のののののIDirectDraw::CreateClipper

DirectDrawClipperƒƒƒƒƒƒ‚□□‚□IDirectDrawClipper::SetHWndƒƒƒƒ„„„ƒƒƒƒ
ƒƒ‚ƒƒƒƒƒ‚Š˜•‚□IDirectDrawSurface::SetClipper

D3DAppCreateFromHWND
の
IDirectDraw::CreatePalette
IDirectDrawSurface::SetPalette
の

IDirectDraw::CreateSurface のZ
IDirectDrawSurface::AddAttachedSurface Z
Z
IDirectDrawSurface::GetSurfaceDesc

IDirect3DDevice

IDirectDrawSurface::QueryInterface
IDirect3DDevice::EnumTextureFormats
の CreateD3D**App**
のの

の

Step 3: の

の
D3DAppCreateFromHWND のStep 5: のの

Direct3D の
D3DAppCreateFromHWND TRUE

D3DAppCreateFromHWND ののの Direct3D DirectDraw
FALSE

Step 3: の

D3DAppCreateFromHWND の 3 AfterDeviceCreated
D3dmain.cpp
Aft*erDeviceCreated* *Direct3D*
D3DAppCreateFromHWND

IDirect3D::CreateViewport
IDirect3DDevice::AddViewport **Direct3D**
D3DVIEWPORT の のの の
IDirect3DViewport::SetViewport の

AfterDeviceCreated **InitView** **InitView**
D3dmain.cpp の InitScene D3dmain.cpp
InitView ののStep 4: の

InitView のののの CleanUpAndPostQuit AfterDeviceCreated
CleanUpAndPostQuit Step 8:

Step 4: の

D3dmain.cpp の
InitView の
Oct1.c InitView の

InitView IDirect3**D::CreateMaterial**
IDirect3DM**aterial::SetMaterial**
IDirect3DMaterial::GetHandle IDirect3DViewport::SetBa**ckground**

InitView ののの
InitView MAKE_MATRIX
MAKE_MATRIX D3dmacs.h の

```
#define MAKE_MATRIX(lpDev, handle, data) \

   if (lpDev->lpVtbl->CreateMatrix(lpDev, &handle) != D3D_OK) \

      return FALSE; \

   if (lpDev->lpVtbl->SetMatrix(lpDev, handle, &data) != D3D_OK) \

      return FALSE
```

MAKE_MATRIX IDirect3DDevice::CreateMatrix
IDirect3DDevice::SetMatrix

InitView
D3DEXECU**TEBUFFERDESC** の**IDirect3DDevice::CreateExecuteBu**ffer

IDirect3DExecuteBuffer::Lock

InitView D3dmacs.h OP_STATE_TRANSFORM
STATE_DATA Step 5: の¥ののの

InitView
IDirect3DExecuteBuffer::Unlock
IDirect3DExecuteBuffer::SetE**xecuteData**
IDirect3DDevice::BeginScene IDirect3DDevic**e::Execute**
IDirect3DDevice::EndScene
IDirect3DExecuteBuffer::Release

InitView IDirect3D::CreateMaterial **D3DMATERIAL** の
IDirect3DMaterial::SetMaterial
IDirect3DMaterial::GetHandle ののD3DLIGHTSTATETYPE の
D3DLIGHTSTATE_MATERIAL

InitView のの D3DVERTEX の D3DVALUE
D3DVALP x
D3DRMVectorNormalize

のInitView のののの

InitView Oct1.c のD3DLIGHT
IDirect3D::CreateLight IDirect3DLight::SetLight
IDirect3DViewport::AddLight

Step 5: のの

D3dcalls.c D3DAppISetRenderState の
ののの
D3DAppCreateFromHWND D3da**pp.c** **D3DAppISetRenderState**

D3DAppISetRenderState のの
D3DAppISetRenderState

D3DAppISetRenderState **D3**DEXECUTEBUFFERDESC
D3DEXECUTEDATA の
IDirect3DDevice::CreateExecuteBuffer
IDirect3DExecuteBuffer::Lock

```
BOOL D3DAppISetRenderState()

{

D3DEXECUTEBUFFERDESC debDesc;

D3DEXECUTEDATA d3dExData;

LPDIRECT3DEXECUTEBUFFER lpD3DExCmdBuf = NULL;

LPVOID lpBuffer, lpInsStart;

size_t size;
```

```
//

size = 0;
size += sizeof(D3DINSTRUCTION) * 3;
size += sizeof(D3DSTATE) * 17;
memset(&debDesc, 0, sizeof(D3DEXECUTEBUFFERDESC));
debDesc.dwSize = sizeof(D3DEXECUTEBUFFERDESC);
debDesc.dwFlags = D3DDEB_BUFSIZE;
debDesc.dwBufferSize = size;

LastError = d3dappi.lpD3DDevice->lpVtbl->CreateExecuteBuffer(
   d3dappi.lpD3DDevice, &debDesc, &lpD3DExCmdBuf, NULL);

LastError = lpD3DExCmdBuf->lpVtbl->Lock(lpD3DExCmdBuf, &debDesc);
memset(debDesc.lpData, 0, size);

lpInsStart = debDesc.lpData;
lpBuffer = lpInsStart;
```

IDirect3DDevice::CreateExecuteBuffer のの d3dappi.*lpD3DDevice* *Direct3DDevi*ce のdebDesc D3DEXECUTEBUFFERDESC の lpData

D3DAppISetRenderState の **D3DAppISetRenderState** **OP_STATE_DATA** の PUTD3**DINSTRUCTION** のの **SDK** **D3dmacs.h** の

```
#define PUTD3DINSTRUCTION(op, sz, cnt, ptr) \
   ((LPD3DINSTRUCTION) ptr)->bOpcode = op; \
   ((LPD3DINSTRUCTION) ptr)->bSize = sz; \
   ((LPD3DINSTRUCTION) ptr)->wCount = cnt; \
   ptr = (void *)(((LPD3DINSTRUCTION) ptr) + 1)
#define OP_STATE_RENDER(cnt, ptr) \
   PUTD3DINSTRUCTION(D3DOP_STATERENDER, sizeof(D3DSTATE),
cnt, ptr)
```

PUTD3DINSTRUCTION の D3DINSTRUCTION の
OP_STATE_RENDER の PUTD3DINSTRUCTION の 1
D3DOP_STATERENDER D3DOPCODE のの2 の
D3DRENDERSTATETYPE の D3DSTATE の

D3dmacs.h STATE_DATA
の D3DSTATE の D3DRENDERSTATETYPE
の

```
#define STATE_DATA(type, arg, ptr) \

   ((LPD3DSTATE) ptr)->drstRenderStateType =
(D3DRENDERSTATETYPE)type; \

   ((LPD3DSTATE) ptr)->dwArg[0] = arg; \

   ptr = (void *)(((LPD3DSTATE) ptr) + 1)
```

D3DAppISetRenderState の OP_STATE_RENDER
STATE_DATA 14
d3dapprs D3dapp.h D3DAppRenderState

```
OP_STATE_RENDER(14, lpBuffer);

   STATE_DATA(D3DRENDERSTATE_SHADEMODE,
d3dapprs.ShadeMode, lpBuffer);

   STATE_DATA(D3DRENDERSTATE_TEXTUREPERSPECTIVE,

      d3dapprs.bPerspCorrect, lpBuffer);

   STATE_DATA(D3DRENDERSTATE_ZENABLE, d3dapprs.bZBufferOn
&&

      d3dappi.ThisDriver.bDoesZBuffer, lpBuffer);

   STATE_DATA(D3DRENDERSTATE_ZWRITEENABLE,
d3dapprs.bZBufferOn,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_ZFUNC, D3DCMP_LESSEQUAL,
lpBuffer);

   STATE_DATA(D3DRENDERSTATE_TEXTUREMAG,
d3dapprs.TextureFilter,

      lpBuffer);
   STATE_DATA(D3DRENDERSTATE_TEXTUREMIN,
d3dapprs.TextureFilter,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_TEXTUREMAPBLEND,
d3dapprs.TextureBlend,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_FILLMODE, d3dapprs.FillMode,
lpBuffer);

   STATE_DATA(D3DRENDERSTATE_DITHERENABLE,
d3dapprs.bDithering,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_SPECULARENABLE,
d3dapprs.bSpecular,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_ANTIALIAS,
d3dapprs.bAntialiasing,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_FOGENABLE,
d3dapprs.bFogEnabled,

      lpBuffer);

   STATE_DATA(D3DRENDERSTATE_FOGCOLOR, d3dapprs.FogColor,
lpBuffer);


OP_STATE_RENDER    STATE_DATA ∅OP_EXIT            D3DOPCODE
          D3DOP_EXIT                             PUTD3DINSTRUCTION


OP_STATE_LIGHT(3, lpBuffer);

   STATE_DATA(D3DLIGHTSTATE_FOGMODE, d3dapprs.bFogEnabled ?

      d3dapprs.FogMode : D3DFOG_NONE, lpBuffer);

   STATE_DATA(D3DLIGHTSTATE_FOGSTART,

      *(unsigned long*)&d3dapprs.FogStart, lpBuffer);

   STATE_DATA(D3DLIGHTSTATE_FOGEND, *(unsigned
long*)&d3dapprs.FogEnd,

      lpBuffer);
```

OP_EXIT(lpB*uffer);*

*D3DApp*ISetRenderState
IDirect3DExecuteBuffer::Unlock
IDirect3DExecuteBuffer::SetExecuteData
IDirect3DDevice::BeginScene IDirect3DDevice::Execute
IDirect3DDevice::EndScene

*LastError = lpD3DEx*CmdBuf->lpVtbl->Unlock(lpD3DExCmdBuf);

```
memset(&d3dExData, 0, sizeof(D3DEXECUTEDATA));
d3dExData.dwSize = sizeof(D3DEXECUTEDATA);
d3dExData.dwInstructionOffset = (ULONG) 0;
d3dExData.dwInstructionLength = (ULONG) ((char*)lpBuffer -
   (char*)lpInsStart);
lpD3DExCmdBuf->lpVtbl->SetExecuteData(lpD3DExCmdBuf, &d3dExData);
LastError =
   d3dappi.lpD3DDevice->lpVtbl->BeginScene(d3dappi.lpD3DDevice);
LastError =
   d3dappi.lpD3DDevice->lpVtbl->Execute(d3dappi.lpD3DDevice,
      lpD3DExCmdBuf, d3dappi.lpD3DViewport);
LastError = d3dappi.lpD3DDevice->lpVtbl->EndScene(d3dappi.lpD3DDevice);
```

D3DAppISetRenderState
IDirect3DExecuteBuffer::Release

```
lpD3DExCmdBuf->lpVtbl->Release(lpD3DExCmdBuf);
return TRUE;
}
```