

## □ Part A

Microsoft( DirectX( 3  
fffffš”fff

TM TM , f f f f f f , < [ ] , , , , , [ ] • , [ ] —  
[ ] , , , • [ ] , , , , , , , [ ] , , [ ] • , ' , % Ž — [ ] —  
 , , , “ “ < Š “ , , , , , • Ž — , , , • “ , , , , , , , ,  
Microsoft , , f f f f f f , < , , , , f f f f f f , “ , , , , ‘ , “ • ’ Ć , , , “ —  
Ć • Ž , , , , , f f f f f f , Microsoft , — , , , % , , , “ < • ’ Ć , , , “ — Ć , % , —  
 , , , , , , , ,

(1996 Microsoft Corporation. All rights reserved.)

Microsoft ActiveMovie Direct3D DirectDraw DirectInput DirectPlay DirectSound DirectX MS-DOS Win32 Windows ,,,Windows NT, • Microsoft Corporation,• ,,,,‘ ,,,, •,,,

., ' , •-, %0Ž-, ŠŽ, •,,,

‘5  
 Direct3D,Š—  
 µDirect3D,Š— §  
 Direct3D,,, §  
 Direct3Df fffff §  
 3Dffffffff,Š‘< §  
 fff fff, “‰ §  
 •Žf f,Š— §  
 •Žf f,,, §  
 ffffffffff: Direct3D,•Žf f fffffff §

<b>Direct3D,•Žf□f,ff□ffff</b>	<b>§</b>
’□f□f,Š—.....	§
’□f f,,,,.....	§
fffffffff: Direct3D,’ f f fffff.....	§
Direct3D,’ f f,ff ffff.....	§

Direct3D,Š—  
Direct3D,,,,  
Microsoft,Š'3Dffffffff fff fff,, Direct3D™,,OpenGL,ffff fff f  
ffffffff fff ffff API ,Š,,,,,  
Ž, , Windows,ffffffff ffff, Win32®  
API,< ,,ffff□fff□,,‘Windowsffff□,,,f□ffff,,ŠŒ,Ž,,,,,,  
..... μ §

Direct3D  
Direct3D, Œ Ž—,,,,,f ffffffff f PC ,ffffffff,fff fff ,Ž—  
,,f f,,,Microsoft,'<,□ffffffff,ffffffff,3D< ,, Direct3D, “ “ ,^,ffff,  
,,,  
Direct3D, Š”Ž,•—  
,,APIf ff,ffff”^‘ ,’< f ffff fff ,^”“,ffff fff,<,,□Direct3D,,,  
□f fff ffff f, ‘Œ,3DffffffffŠ<,,,ff f,,,,,, fff f f,ffff,ffff  
,3DffffffffŠ<,’%o,,,,,Š’,,,, ffff,,,,,fff,,•—  
,,,,, ffff f ,,,,,,ffff□f□ffff□,%o““ , ,

Direct3D,fff fff 3D  
ffffffff f ff,Š‘fff,, fffffff f f, ‘,ffffff fffffff •Š — fff  
—  
, f ffff fffff f,,’ “,ffff,ŽŒ,,□APIf□ff,,□“,,□fff,•Žf□fAPI,□'f  
ff,□f□fAPI,,,□Direct3D,Ž—  
„3Df ffff fffff f,ffff,,ffff,ff f,, Direct3D,Š‘,ff fff, f ffff,  
Direct3D,,,3Dffffffff fffffff,^,,,‘, ‘%o  
,,,,,, Direct3D,□Zfffffffff□ffffffffffff□ffff□fffffffff□fffffffff  
ffffff Š,f f,,ffffff fffff,, 3Df ffff fffff f,Š'ffffff”—  
, ,,,,,, Direct3D,‘DirectXfffff ,Š‘“ ,,,,,, fff fffff 2D,f□f□ff  
□ff□f,,3Dffffffff fffff,,Š'<“ ,,,, ,,,,,, fffffff,ffff ffff, f f  
ff,2D,3D,ffffffff,Ž—,,,,,,

Direct3D,ffffffff•—  
,, •Žf f,‘ f□f,,,□,,,^,,,f f,, ,•Žf f, ffff fff,ffffff f f,•Ž,,  
fffAPI,, ,‘ f f, ffff fff,—Ž“Ž ffff,f f, —,,’fffAPI,,  
,, , Direct3D,’ f f,•Žf□f,,,□—,,,,□Direct3D,’——  
,,,,,f□ffff□fffffffffff□'□%o□,ffff fff,,,,Ž,^,  
•.....•Žf f  
•.....’ f f  
•.....f□ffff□fffffffffff□'□%o□,ffff fff

•Žf f  
Direct3D,•Žf□fAPI,□3Dffffffff,‘ ,3Dffffffff,□—,,,□Œ,,,,□•Žf□f,—  
,,□Š‘, Windowsffff□fff,,3D<”’%o, ,3Dffff□fff,□□,—  
^,□,,,,□•Žf□f,‘,□,,,ffffffffffff□f□ff□f□fff□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',ff□fff□fff,ffffff Windows

の 3D ののの

の

のの3D 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

DirectDraw  
DirectDraw  
DirectDraw 2D 3D の Windows  
DirectDraw の  
の DirectDraw 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  
の Win32 Win32 OpenGL  
の OpenGL OpenGL

OpenGL の

の Direct3D、Š,,f ffff fff fff,’<, Direct3D  
API,’,%Š,3Df ffff OpenGL,•—,,, -, Ą,ff f,,,, ,——  
,,,,,,,,,—’,,,,

Direct3Df fffff

,, ,, Direct3Dfff fff,,‘,DirectXfff fff,fff ffff ffff ,,ffff f f  
fff,,ŠĀ,,,, ‘ “, •,< ,,,, ,,,,^%,ffffff,,, -,,

- Direct3D,ffff
- 
- 
- 
- 
- 
- 
- 

• Direct3D

Direct3D の

Direct3D Windows  
3D  
3D の

Direct3D

3D

の

Direct3D の DirectDraw  
3D の

DirectDraw

の 2D の

の 3D の

Direct3D 3D  
の の

Direct3D  
の の

Direct3D の

Direct3D ffffff, ffff,

$$\begin{array}{ccccc}
 \text{Direct3D} & \text{DirectDraw} & & & \text{DirectDraw} \circ \text{COM} \\
 & \text{Direct3D} \circ \text{COM} & & & \\
 & & \text{Direct3D} & \text{DirectDraw} \circ & \text{DirectDraw} \\
 \circ \text{3D} \circ \text{3D} \circ & & & & 
 \end{array}$$

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

2D 3D の

のののの Direct3D

§

のの

§

DirectDraw Direct3D  
DirectDraw

”,,,□,,□Direct3D,-,□,,,□ffff□ffff□f□f,“□,,,ffff□ffff,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  
dwMaxBufferSize  
,,,□□□^”“,□□“,Ž□fff 64K のの

D3DDEVICEDESC の  
Ž—

		Direct3D の	
の			4 の
の		3D の	
3D	2D		
の			
の			
の D3DTLVERTEX		D3DTLVERTEX	
		D3DLVERTEX の	
		D3DLVERTEX	
D3DVERTEX の			
		D3DTRANSFORMDATA	
の API			
の 3D			
		$fff, ..., ffff, \cdot Z,$	
		D3DLIGHTINGELEMENT	
の			
の			
		RGB の D3DLIGHTDATA の	
D3DLIGHTINGELEMENT の			
の ”	”		
の		RGB の	
の			
の 0 の		1 の	
§			

ののののののの 3/4 ののの 4 の 1 ののの□□  
□□□□□□□□□□□□□□□□□□□□□□□□□□□□  
□□□□□□□□□,□

**fff,,,RGBff**□,□**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;

ⒺⒺ,fff, D3DLIGHTTYPE

—ⒺⒺ,fff,,,, ,,, D3DLIGHT\_DIRECTIONAL D3DLIGHT\_POINT D  
3DLIGHT\_PARALLELPOINT D3DLIGHT\_SPOT D3DLIGHT\_GLSPOT,  
,,,,,, ,,—ⒺⒺ, D3DLIGHT “,^•,, , “, ‘,D3DCOLORVALUE

“,Š,,,, ⒺⒺ, Ž’,,,, —  
,—‘,’”^, ~”0,,1,,,, fffⒺⒺfff,’ Ž,“Ž,Š,,,,,

§

Š—‘,,0,,1,”Š,’Ž’%”, f f fff,,, —Ⓔ%o

,Š’,,,,,, D3DLIGHT “,Ž,•Ⓔffff, fff,,ⒺⒺ,,,•Ⓔ,•,,,, ,ffff, —

ⒺⒺⒺ,,,• ,—,,,,, ffff,,,f ff •,Ž’, ,,,ffff, Ⓔ ,f ff —

,,,fff •,Š,, ,,, ffff,f ff •,Š,,,,, Ⓔ—“,fff, —Ⓔ%o

,Ž,,,,, ffffffff,ffffff,, ”^fff f, —

Ⓔ,ⒺⒺ,”^Ž,,,, ,”^Š•,,“,, Ⓔ,“,,,,, Ⓔ,<,2ŽⒺ Ⓔ ,,,•%o

,, Ⓔ,“,,,,’“,,ⒺⒺ,,,<—,d,,, Ž,•Ž, —,,

§

D3DLIGHT□“,Ž,fff□dvTheta,,,dvPhi□,□,,,□ffffff,—,,%o

,Ⓔ□,Š“,’<,,ⒺⒺ□Ⓔ□□dvFalloff



の  
の D3DLIGHTDATA

の

IDirect3DDevice::Execute のの

の(0, 0)の ( -1, -1)の 0 1  
のののののののの

Direct3D の

のDirect3D の

- 
- 

IDirect3DRM::CreateDeviceFromSurface DirectDraw  
Direct3D  
DirectDraw の DirectDraw  
の SDK のDirect3D  
WM\_ACTIVATE Direct3D 8 の DirectDraw の  
IDirectDrawPalette::GetEntries

のの

D3DPAL\_FREE

D3DPAL\_READONLY

D3DPAL\_RESERVED

のWin32 の PALETTEENTRY の peFlags のD3DRMPALETTEENTRY の  
D3DRMPALETTEFLAGS RGB

の  
RGB の

D3DTLVERTEX のspecular の  
RGBF "F" "fog"の F

の  
の  
の

の  
の

§  
•Ž□fff□f□f,,□Ž□fff□f  
§

の e 2.71828,,, fff,“-,•,Š ,,,  
fff,’, ,,,, fffffff,Œ,,,,,  
ffff fff,Ž fff f f,Ž—,, fff,—“0.5,,,,,, Ž,—Ž,,,, fff,,0.8,←  
,,^’,fff,’0.6703,,,  
§

ff f,ff f f f fff  
Direct3D, Œ , ‘, ’,,,,, Ž ffff’,f f,%o ,, ffff fff, ffff,f f,ff  
ffff,,,,ŽŽ,,‘, ‘,ffffff,, D3DSTATE “,, ,,f fffff,Ž ,,Ž,,,—Œ,  
Š,,,, D3DTRANSFORMSTATETYPE,•Šfff f, ‘, ’,, ,, D3DLIGHT  
STATETYPE, -fff f, ‘, D3DRENDERSTATETYPE,fff —  
fff f, ‘, ’,,  
,,,, ‘, “,Ž, —Ž,BOOL’,fff,Ž,,,, ,,fff,TRUE, ,^ ,• , ,,,  
ffff fff, D3DSTATE\_OVERRIDEfff,Ž—,, “,Ž, — ‘,-  
Œ,,,,,, ,<”,,,,ffff fff, Ž ffff, —  
, ffff, ‘,•,,,,,“ ,• ,,,,,, Direct3D,•Žf f,, ff f f f fff,—  
—,Ž ,,,fff,,, ff f f f fff,—,,, Ž ffff,Š‘, ’,•—  
, ,,,, ,,,, ffff,ffffff,ff f,ffffff,’Š,,, •Žf fAPI,ff f f f fff  
,—,,  
ffff□fff,□  
ff□□ff□ffff□f□f,fff,,,ffffff,,,,,D3DSTATE\_OVERRIDEfff,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  
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

Direct3D

Direct3D の 3D ののののののの

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

Direct3D の API Direct3D

3D の

の 3D ののののののの Direct3D の **ff** , **Ž** , , , ,

3D Direct3D の

3D

3D の 2 の の  
の x の y の z  
の の x y

の  
の z の

の Direct3D の

- Direct3D の
- U V

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

3D  
3D の

- の の
- の
- 

のの4 4 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 API

のDirect3D

の(x, y, z) (x', y', z')

§

の

□, “,,,,,,□,fff

の(x, y, z) x

(x', y', z')

§

のy

§

のz

§

ののの $\theta$

$$\mathcal{O}(x, y, z) = x \cdot y \cdot z \mathcal{O}(x', y', z')$$

§

Direct3D の 3 のののの

の **Direct3D**

- 
- 
- 
- 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

ffffff,‘□,,,,,’

の

§

§

の

∞

□□□□□  
のDirect3D

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  
□□□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

□%, , ☒, , •%, , , , , , , , □

☒☒, "Š□, ž, , , □, , , , žŠ☒, , , , ' " , < - , " , ^ , , , , , □

§

ffff, □' "v0□v1□v2, -, , □□, žŠ☒, •%, , □2"-, žŠ☒, •%, , □' "v0□v2□v3, ž-, , □

"Š□, □☒

D3DTRIANGLE の wFlags

□□□□□□□□□□ ffff

Direct3D, , □' " , ^ ' , ☒, , ž, , , , □fffff' , Š' " , □^ ' , ž, ffff☒☒, , ž, -

□fffff□fffff, □•□, , , □, , , , < □, , , □•žf□f, , □, , , , ' , D3DRMVERTEX□' \ , Š" , , , □□



[illegible]
$$\begin{array}{ccccccc} \mathcal{O} & & & & & & \\ & 2 & & r_2 \mathcal{O} & & 1 \mathcal{O} & r_2 \end{array}$$

①

Direct3D の D3DRMQuaternionFromRotation  
D3DRMQUATERNION  
D3DRMQuaternionMultiply  
D3DRMQuaternionSlerp

### D3DRMQuaternionFromRotation

D3DRMQuaternionMultiply

D3DRMQuaternionSlerp

D3DRMVectorAdd

D3DRMVectorCrossProduct

D3DRMVectorDotProduct

D3DRMVectorModulus

D3DRMVectorNormalize

D3DRMVectorRandom

D3DRMVectorReflect

D3DRMVectorRotate


D3DRMVectorScale

D3DRMVectorSubtract

Direct3D の DirectX  
の 53 の

53 の

の

3D  


))))

Direct3D

•  $\mathcal{O}$

- $\mathcal{O}$
- $\mathcal{O}$
- 

Direct3D	の
RGB の	の

- $\mathcal{O}$
- 
- $Z$
- 

```

IDirect3DDevice::Execute                                ffff,—
,,□,,,fffff,□,,

```

 $\mathcal{O}$ 

IDirect3DDevice::Execute ④

	D3DSTATUS 0
00	D3DOPCODE 0
D3DOP_BRANCHFORWARD	
	Direct3D 0

 $\mathcal{O}$  $\mathcal{O}$

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 ffff fff,•-,Ž ,,,,,,,

D3DTRIFLAG\_STARTFLAT(len)

Œ ,ŽŠŒ,Ž, ,,,, ffff, “Š , Œ,, Œ,‘,ŽŠŒ,len,Ž, ,,Ž, ,,,,,,

D3DTRIFLAG\_ODD,D3DTRIFLAG\_EVEN

ffff,□ŽŠŒ,,,,,□,,’“,,, f□f,,,,□‘,,,,“,fffff

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
- 

ffff,fff fff,16~%0,,,

- 

,,,ffffff fffff ffff, , ,“^,  
, fffff,ff fff,Ž’,,,,,,, ,,,  
,,,ffffff,”,,, ”Ž—  
“,\*, ‘,ffff fff,, ,,,f f,,  
,^ ,ffffff,\*—  
,, ,,,œ,”Ž,,,ffffff,, ,,,,,,”

Ž,

- 
- の256×256 の1 の
- 

のの  
□□□□□□ のの

のののの  
□□□□□□□ のののの  
f□f□ff,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 の の の の SDK の Direct3D の  
の の の の D3dtest.exe の の の の の

Direct3D の の-systemmemory  
の の fail

DirectDraw WIN16  
WIN16fff, □GDI, USER,, ffff, ffff%o, □ IDirectDrawSurface2::Lock  
IDirectDrawSurface2::Unlock の の Windows  
IDirectDrawSurface2::GetDC IDirectDrawSurface2::ReleaseDC の の

**D3DTEXTUREBLEND** の

D3DTBLEND\_COPY

8 の 16 16  
の 8 の 2 8 の の の  
D3dtest.exe  
の

の Direct3D の 3D Microsoft  
3D Direct3D の API

Direct3D DirectDraw DirectDraw  
DirectDraw Direct3D の の IDirectDraw::QueryInterface





DD



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::QueryInterface* の Direct3DRMDevice 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
                  •—
                  ,,,, fffŽ•Ž CLSID ,Ž,,,, IDirect3DRM::CreateObjectffff,Œ, ,,, ”
                  —ffffff, ,,,,,,, ,,‘,, Šfff ffff,,,~%o, ffff,Ž—,,
                  ”—fff ffff
                  ffff

```

```
IDirect3DRMDeviceArray
```

**IDirect3DRM::GetDevices**

**IDirect3DRMFaceArray**

```
IDirect3DRMMeshBuilder::GetFaces
```

```
IDirect3DRMFrameArray
```

```
IDirect3DRMPickedArray::GetPick
```

```
IDirect3DRMFrame::GetChildren
```

```
IDirect3DRMLightArray
```

```
IDirect3DRMFrame::GetLights
```

IDirect3DRMPickedArray

IDirect3DRMViewport::Pick

IDirect3DRMViewportArray

IDirect3DRM::CreateFrame

IDirect3DRMVisualArray

IDirect3DRMFrame::GetVisuals

fffff, Ź ffff,0,,,,, ,,,,,,
ffff fff, f f,f ffffffff,,,,, Ź ffff,•Ž,,•—
,, ‘,ffffff,Ž ffff, Ź““,ffff,Š—,, ffff fff, —
,,‘,f f,ff f f ffff,%o•,,,,,, ffff fff,ff f f,%o•,,, ffff,Ž““,ff
f,Ž ffff, —,, —““,,,ff f f,ffff,%o
,, ,,, ffff fff,ffff,%o•,,,ff f f,%o•,,,,,, ,,, ffff,%o•,,,,, “
—,ff f f,%o•,,,,,,
Žffffff,ffffff□ffffff,Ž□ffff,□ff□f,%o
,,,,,‘,,,□IDirect3DRMFrame::AddChildffff,Ž—
,,□,,□ffffff,,•,□,Žffffff,^“,,,□ffff,Ž““,Ž□ffff,‘□,,□
ffff□fff,ffff

のののの

IDirect3DRM

IDirect3DRM の
Direct3DRM ののIDirect3DRM
IDirect3DRM の

- *fff*□*ff*
- 
- face
- *ff f*
- Š%,,,,,~”*ffffff*
- Œ
- *fffff*
- *ffff,ffff fff*
- %o%o
- *fffff*
- *f f fffff*
- 
- 

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 *fff*□*f*,<−  
 ,□•Ž,*fff*□*fff*,□□,Š−%  
 ,,,□*ffff*□*fff*,□IDirect3DRMAnimationSet::AddAnimation*ffff*,−  
 ,,*fff*□*fff*,*fff*□*fff*□*fff*,’%  
 ,,,,,,□*fff*□*fff*,□□,,,,□IDirect3DRMAnimationSet::DeleteAnimation*ffff*,  
 Ž−,,□*fff*□*fff*□*fff*,□IDirect3DRMAnimationSet::SetTime*ffff*,Œ,□,,,,,□Œ,,,,□

```

Š~•,,,,,□□IDirect3DRMAnimation□,,,□IDirect3DRMAnimationSet□fff□fff,
Ž□,,,,□
IDirect3DRMDevice,IDirect3DRMDeviceArrayfff□fff
ffffff,,,,,□—ŒŽ,□□—ffff,Š~,,,,,,□fff□ffffff,□ffffff,□—
□,%Žffffff,•,,,,□
ffffff,“□,□Ž’,,,□
    ,,,
•Žf f, fff f,ffffff ,,,ffff fff fff,’ fffffff,,ffff,ff f,,
Š~ •,,,, IDirect3DRMDevice ,Ž ,,,,
,,ffffff,, Direct3Dff f, —ffff,•Ž,, ,—Œ,ffffff,,, —,,
• fffff
• ff fff
•

```

```

IDirect3DRMDevice::SetQuality
IDirect3DRMMeshBuilder::SetQuality のIDirect3DRMDevice::GetQuality
IDirect3DRMMeshBuilder::GetQuality

```

RGB

IDirect3DRMDevice::GetColorModel

RGB	,•□,ŒŒ
の	8 16 24
32 の	
24 の	

の

の □f,,,のののの RGB の

の

の

RGB	8 16 24 32 の
	8 の

RGB

Direct3D の IDirect3D::EnumDevices  
の の

IDirect3D::FindDevice

WM\_PAINT WM\_ACTIVATE

WM\_MOVE

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::GetVertex *ffff,Ž*—

IDirect3DRMFace

IDirect3DRMFrame IDirect3DRMFrameArray

の の の の の

の の の

の の の の の の の の の

の の 1 NULL IDirect3DRM::CreateFrame

*Direct3D の 3D*

IDirect3DRMFrame

の の

•

•

•

•

の

の のの  
IDirect3DRMFrame::AddChildffff,, ,,,,'ff f,'%  
,,,,,, Žff f,Š', ,,,IDirect3DRMFrame::DeleteChildffff,Ž—  
,, Žff f, ff f,Ž“, ,,, IDirect3DRMFrame::GetChildren,IDirect3DRMF  
rame::GetParentffff,Ž—,,  
ff□f,□',ff□f,ffffff□ffffff,,,'%,, —,,Š' ' , f□f““,“,%,“,—  
,,,,%”,,, ,Š',ffffff,Ĉ,,,□fff□fff,'%,—  
,,, ff□f,Žff□f,“, ,,,,,',—  
,,,□•Žf□f,□Ž□Ž,□%Š',ffff,□,,, □□%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  
**D3DRMCOMBINETYPE** のの の  
IDirect3DRMFrame::GetRotation IDirect3DRMFrame::GetTransform の  
の IDirect3DRMFrame::SetRotation  
IDirect3DRMFrame::Transform  
IDirect3DRMFrame::InverseTransform

の

のの



のDirect3D  
の3D

0 □□□□□□

の

□*f*’,

IDirect3DRMFrame::AddMoveCallback

Direct3DRMFrame::DeleteMoveCallback

のの

ののの

の

IDirect3DRMLight IDirect3DRMLightArray

,, ,,,ffffff, ,, fffffff, , ff  
ffff’,\$CE%,%o€, —,,

CE’ の

ののののの

のの

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 IDirect3DRMLight::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2D

IDirect3DRMMesh IDirect3DRMMeshBuilder のの COM  
IDirect3DRMMesh

IDirect3DRMMeshBuilder IDirect3DRMMesh の  
IDirect3DRMMeshBuilder

**Direct3DRMMeshBuilder**

**Direct3DRMMesh**

の

のの

の

IDirect3DRMMesh

のの

IDirect3DRMMesh::AddGroup

の

のの

IDirect3DRMMeshBuilder IDirect3DRMMesh の  
64 のののののDirect3DRMMesh

**Direct3DRMMeshBuilder API Direct3D API**

IDirect3DRMMeshBuilder::AddVertex

IDirect3DRMMeshBuilder::AddFace

IDirect3DRMMeshBuilder::AddFaces

IDirect3DRMMesh::SetGroupColor

IDirect3DRMMesh::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



IDirect3D::CreateShadow の  
 IDirect3DShadow の  
 IDirect3D::CreateObject  
 IDirect3DShadow::Init  
 IDirect3DTexture  
 のののののの  
**IDirect3DTexture** **DirectDrawSurface**  
 Direct3D のDirect3D の DirectDraw の  
 のDirect3D の  
 D3DRMIMAGE  
 IDirect3D::CreateTexture  
 IDirect3D::CreateTextureFromSurface DirectDraw  
 IDirect3D::LoadTexture  
 Windows の .bmp  
 .ppm

IDirect3DWrap Interface  
 の **IDirect3DTexture**  
**Direct3D** の  
 •  
 • の  
 •  
 •  
 • の

の  
 IDirect3DTexture::SetDecalSize IDirect3DTexture::SetDecalSize の  
 の IDirect3DTexture::SetDecalOrigin  
 IDirect3DTexture::GetDecalOrigin のの[0, 0]

$\mathcal{D}$

IDirect3DFrame  
 IDirect3DFrame::CreateUserVisual  
 IDirect3DFrameUserVisual::Init  
 の

IDirect3DFrameViewport and IDirect3DFrameViewportArray

3D の 2D  
 の  
 ののの IDirect3DFrameViewport  
 ののの  
 •  
 • Ž□‘  
 •  
 •

IDirect3DFrameFrame  
 z y の の

IDirect3DFrameViewport::SetCamera のののの  
 IDirect3DFrameViewport::GetCamera

,3Dfff□f,,,□Ž□‘,’,,fffff,%oŽ□‘,,,□“Ž“%o,,□ffff□fffff□ののの  
 §

のの z D の  
 F ののIDirect3DFrameViewport::SetFront  
 IDirect3DFrameViewport::SetBack IDirect3DFrameViewport::GetFront  
 IDirect3DFrameViewport::GetBack の 2h の h の  
 IDirect3DFrameViewport::SetField IDirect3DFrameViewport::GetField

の A の  
 のh の  
 §

**D3DRMPROJECTIONTYPE**  
**IDirect3DRMViewport::GetProjection**  
**IDirect3DRMViewport::SetProjection**

**3D 2D**

4 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  
 W の

の

P

§

sx sy ox oy [-h -h D] [h h D] の の

f •,ffff,Š,,, IDirect3DRMViewport::Transform,IDirect3DRMViewport:  
 :InverseTransformffff,Ž—,, Ž,—,,, ffff fff, ,,,,ffff,—  
 ,,ffff,ff f,,,,,,,,

/\*

\* 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);
}

```

$\infty$   
*IDirect3DRMViewport::Transform* の 4 の  
 の  
 の 3D の [x y z w] の  
 §

Direct3D  
**3D**

の **2D** の  
*IDirect3DRMViewport::Pick* の

*IDirect3DRMVisual* and *IDirect3DRMVisualArray*

***IDirect3DRMFrame::AddVisual***

の *IDirect3DRMVisualArray*  
*IDirect3DRMVisual COM*

*ffffff,fff,□Direct3DRMMeshBuilder*  
*Direct3DRMTexture*  
*IDirect3DRMWrap*

の  
*IDirect3DRM::CreateWrap*  
*IDirect3DRMWrap* の  
*IDirect3DRMWrap::Apply* *IDirect3DRMWrap::ApplyRelative* の  
*IDirect3DRMWrap::Apply* の *IDirect3DRMWrap::ApplyRelative*  
 の

v z u y  
[0, 0, 0]

IDirect3DRMWrap  
IDirect3DRMWrap

4

- 
- 
- 
- 
- 

D3DRMMAPPING D3DRMMAP\_WRAPU  
D3DRMMAP\_WRAPV の

,fffff□•Š,□'←  
u v の  
∞  
v ∞∞∞∞ u

- fffff fff,fff,,,,,•-fffff f f,, u,,v •,Ž',,,•-, fffff,-  
E,fffff,,, ,, —E,u,,v •',1.0^,,, “(0.1, 0.1),(0.9, 0.9)  
,E, ' , ' , “(0.5, 0.5),'%o,,  
•

- D3DRENDERSTATE\_WRAPU,,D3DRENDERSTATE\_WRAPV,,,,,fff,  
,, fffff,%oŽ1.0, ,,,,%o“E,,, 1.0^ ,fffff •, fff,,,,,,”^,,—  
E,,, fffff •Š, '←  
,fffff fff,,,^,, D3DRENDERSTATE\_WRAPU,fff,,,,,,, “(0.1, 0.1)  
,,(0.9, 0.9),,, ' , ' , “ 0, 0.5 ,'%o,,  
• D3DRENDERSTATE\_WRAPU,D3DRENDERSTATE\_WRAPVfff,—  
•,fff,,,,, fffff,%oŠ' f ff ,,, ffff,• ,,,, 1.0^ ,fffff •,—  
E,,, “(0.1, 0.1),,(0.9, 0.9),,, ' , ' , “(0, 0),'%o,,

—E—^,,,fffff •,—E—^,Ž,,,,, ,,,,“ ,• ,,,,  
^”,ffff fff, fffff,E -,•,^,,,,, %o' ,fff,,,,,fffff fff,fff,, ,,,,—  
,fffff,”•^ ,Ž—,,, , fffff fff, ' ,,,

•—

•-fff,, fffff, ffffff ,fffff,,,ff,,,fffff,-,^',,

Ž,“Ž,,, ffff[x y z],[u v] •,<,,,

u=sux—ou

$v=svy-ov$

,,,ŒŽ,,, s,ffffff,ff fffŒ o,ffffff,Œ“,Ž,,, ffff fff, 1‘,ff fff  
Œ , u,,,v,—Œ,’Ž,,,x,y,0,,1,”^,fff,,,,ffffff,Œ’,,,,,,,,  
%o’  
%o’fff,, fffff,%o’,%o,,•Ž,,,^,, ,,’%o’,,,,,,, fffff,%oŽ,’%o  
,”,, fffff,f fff, ,,ffff,•Œ,,,  
%o , %o’ffffff fff,,,Šffff,Œ%o,Ž,,,,,  
§  
•Œffff,%o’,Ž,Ž,□□ffff,%o“,Š‘,u=0,,“,Ž,□ffff,□ffff[x y z],‘,,ffff,[u  
v]□•Œ

§  
u v z 0 1 Œv  
Œ

[x y 0] x Œu v [x y z]  
z ŒŒz

§  
Œ  
§

u v Œ 0 1  
Œ

,”Ž,,Œ,,,,ffffff •,Š“,,, fff  
fff,Ž ff f,^,Ž“,

”Ž Œ Œ  
Œ Œu v Œ

Direct3D ŒŒ  
Windows Œ Direct3D Œ Œ  
Windows □,,,,,  
ŒŒŒŒŒŒ

§

のの

- Helworld.c

•

•

Windows の

- の
- 3D
- 
- の

•

Helworld.c

3D  
の Direct3D の  
の  
SDK の  
Hello.ppm の  
の  
の Direct3D の  
Sphere3.x  
ffff, - ^,  
SDK の  
DirectX  
3D  
Direct3D の

•

•

- DirectDraw の

の

の Helworld.c  
SDK の Shpere3.x

Helworld.c Helworld.c DirectX  
Hello.ppm の

の DirectX SDK の Globe の —%o  
 ,,,, □ SDK の Direct3D の Globe Rmmain.cpp Helworld.c  
 Rmmain.cpp の C++ C

の の の の の の ”Hello, world!” 3D の

□ □ □ **fff** □ **ffffff**

Direct3D, •Žf □ f □ fffff □ fff, □ fffffff □ ffff 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" SDK ffff, Š,,, Š', Direct3D •Žf f, ffff

//

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

#define INITGUID // ‘, fff’ <, ffff f, ‘,

// ‘<, , , , , , , ,

#include <windows.h>

#include <malloc.h> // memset, ĄE, , , •—

#include <d3drmwin.h>

#define MAX\_DRIVERS 5 // D3D ffff, ‘

```
// ff ff•
```

```
LPDIRECT3DRM lpD3DRM; // Direct3DRMffffff
LPDIRECTDRAWCLIPPER lpDDClipper; // DirectDrawClipperffffff
```

```
struct _myglobs {
```

```
    LPDIRECT3DRMDEVICE dev; // Direct3DRMffff
```

```
    LPDIRECT3DRMVIEWPORT view; // f f,Ž,,,Direct3DRMff f f
```

```
    LPDIRECT3DRMFRAME scene; //
```

```
‘,ffffff,’”,,,fff□□ff□f
```

```
    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; // ∞
```

```
} 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 * lpLightFrame1,
        LPDIRECT3DRMFRAME * lpWorld_frame);
void MakeMyLights(LPDIRECT3DRMFRAME lpScene,
        LPDIRECT3DRMFRAME lpCamera,
        LPDIRECT3DRMFRAME lpLightFrame1,
        LPDIRECT3DRMLIGHT * lpLight1, LPDIRECT3DRMLIGHT *
        lpLight2);
void SetMyPositions(LPDIRECT3DRMFRAME lpScene,
        LPDIRECT3DRMFRAME lpCamera, LPDIRECT3DRMFRAME
        lpLightFrame1,
        LPDIRECT3DRMFRAME lpWorld_frame);
void MakeMyMesh(LPDIRECT3DRMMESHBUILDER * lpSphere3_builder);
void MakeMyWrap(LPDIRECT3DRMMESHBUILDER sphere3_builder,
        LPDIRECT3DRMWAP * lpWrap);
void AddMyTexture(LPDIRECT3DRMMESHBUILDER lpSphere3_builder,
        LPDIRECT3DRMTEXTURE * lpTex);
static void CleanUp(void);

```

Windows の

のHelworld.c

Windows の

- WinMain

- 

WinMain

```

Helworld.c の WinMain      DirectDraw   Direct3D の InitApp
CleanUp      Windows の
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;

// 9

//

```
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) {

        // 2

        //

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

InitApp

```

```

Helwold.c,ŠŠ,Windowsffff,
,ffff,~,ffffffffff,DirectDraw,Direct3D,
,ffff,~,
InitAppŠ,Š,Š,ffff,ffff,Ž,,' ,ffff,ffff,Ž,
,Ž-,Š,fffff,f,Š,Ž,
,,Š,Direct3Dffff,
Š,Š',Š,ffff,Š,EnumDriversŠ,Š,Š,ffff,Š,Š,Š
ffff,Š,Š,Ž,
Ž,ŠDirect3DRMCreateŠ,Š,ŠIDirect3DRM
ffff,Š,Š,ffff,f,f,Š,Š,Š,f,f,Š,Š,ŠIDirect3D
RM::CreateFrame,IDirect3DRMFrame::SetPosition,Š,Š,Š,Š,
DirectDrawClipperffff,ŠDf,f,ŠŠ,Š,ffff,f,f,Š-
,Š',ŠHelworld.c,ŠIDirectDrawClipperffff,Š,Š,DirectDrawCreat
eClipperŠ,Š,ŠIDirectDrawClipper::SetHwndffff,Ž-
,Š,ffff,Ž,ffff,f,f,Š,Š
,Š,f,f',CreateDevAndViewŠ,Š,Š,Direct3Dffff,f,f,f,Š,Š,Š,Š,Š
,Š,ffff,f,f,f,Š,Ž,Š,
Direct3Dffff,f,f,f,ŠŠ,ŠŠ,Š-,ŠDf,f,Š,Š,Š,Š,Š,Š
-,MySceneŠ,Š,Š,MySceneŠ,Š,Š,ffff,f,f,f,Š,Š,Š
ŠŠ,InitAppŠ,Š,Š,ŠŠ,~,ffff,Ž,Š,Š,Š
////////////////////////////////////
//
// InitApp
// fffff,Š,fffff,ŠŽ,Š,Š,ffff,
// ŠŠ,Š
//
////////////////////////////////////

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

    // fffff,Š,fffff,f,~,Š,Š

    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;

    // ff,f,ŠŠ,Š

    memset(&myglobs, 0, sizeof(myglobs));

```

```

// fffff,00,,0

win =
    CreateWindow
    (
        "D3DRM Example",           // fffff0fff
        "Hello World (Direct3DRM)", // fffff0
        WS_VISIBLE | WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU |
            WS_MINIMIZEBOX | WS_MAXIMIZEBOX,
        CW_USEDEFAULT,             // 0$0•
        CW_USEDEFAULT,             // 0$0•
        300,                       // 0$,•
        300,                       // 0$,0,
        NULL,                      // 0fffff
        NULL,                      // fff0fffff
        this_inst,                 // fffff,ffffff0fffff
        NULL                       // 00fff0f
    );
if (!win)
    return FALSE;

// 00,ffffff,ffff",,,fff0,<%,,,0

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

// D3Dffff,-<,0,,,,``,,,0

if (!EnumDrivers(win))
    return FALSE;

// D3DRMffffff,D3DRMffffff,00,,0

lpD3DRM = NULL;
Direct3DRMCreate(&lpD3DRM);

// fff00f0f,ff0f,fff0ff0f,00,,0

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));

// DirectDrawClipperffffff,00,0fffff,$~•,,0

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

// `",,,D3Dffff,-,,D3DRMffff,00,,0

GetClientRect(win, &rc);

```

```

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

        // ffffff,,,f□f,□□,,□

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

        myglobs.bInitialized = TRUE; // □Š%Š-

        // fffff,•Ž,,□

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

        return TRUE;
    }

ffff□fffff□ffff□ff
Helworld.cffff,fff□fffff□ffff□ff,"□,'□,,,□Ž□,□,,ffff,f□f,"-
,`,Ž,•,,,ffff□ffff,,,□
fffff□fff□ff,□WM_DESTROYfff□f,Ž,Ž,,CleanUpŠ□,☉,□,□
,,WM_ACTIVATEfff□f,Ž,Ž,,,,,□fffff□ffff□ff, IDirect3DRMWinDevice,Ž`,□ff
fff,fffff□fffff,□,□,,•Ž,,,,, IDirect3DRMWinDevice::HandleActivatefff
,☉,□,,□,□"-
,□WM_PAINTfff□f,%",,□fffff□ffff□ff, IDirect3DRMWinDevice::HandlePaintff
ff,☉,□,□
////////////////////////////////////
//
// WindowProc
// fff□fffff,fff□f□ffff
//
////////////////////////////////////

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:
        {

            // ,,ffff□f,□-,,□fffff☉-,D3DRMfffff□fffff,

```

```

// □□,,□

LPDIRECT3D3DRMWINDEVICE 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);

    // ,,ffff□f,□-, ,□ffffffE-, D3DRMffffff□ffff,
    // □□,,□

    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

- EnumDrivers

- enumDeviceFunc
- BPPTtoDDBD

## EnumDrivers

## EnumDrivers      InitApp ④

```
Direct3D COM ////////////////////////////////////// DirectDraw ////////////////////////////////////// EnumDrivers
DirectDrawCreate ..... DirectDraw ..... QueryInterface ..... C
QueryInterface ..... IDirect3D ..... // EnumDrivers
//////////////////////////////////////
// EnumDrivers
// -E,D3Dffff,<,□,,,,'\',□
// 
//////////////////////////////////////
static BOOL
EnumDrivers(HWND win)
{
    LPDIRECTDRAW lpDD;
    LPDIRECT3D lpD3D;
    HRESULT rval;

    // DirectDrawffffff,□□,□fff,<-,,Direct3D
    // fff□ffff,-,□,,,□

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

    // enumDeviceFunc,fff`f□,□Š%,,,,CurrDriver,
    // -1,□',□fff,<,□

    myglobals.CurrDriver = -1;
    lpD3D->lptbl->EnumDevices(lpD3D, enumDeviceFunc,
        &myglobals.CurrDriver);

    // □,,,,-E,ffff,,,,,,,,•□,□
```

```

if (myglobs.NumDrivers == 0) {
    return FALSE;
}

lpD3D->lpVtbl->Release(lpD3D);
lpDD->lpVtbl->Release(lpDD);

return TRUE;
}

enumDeviceFuncf□ffffš□
enumDeviceFunkš□,□D3DENUMDEVICESCALBACKE,f□ffffš□,,□D3DENUMDEVICESC
ALLBACKE,fff□ffffD3dcaps.h,'<,,,,,□ffff,,,š□,□ffff□f,,,,šDirect3Dfff
f,ž•ž,-'□,,,,f□ffff,ffff□f,,,ffff,"-','',□
f□ffffš□,□D3DDEVICEDESC□`\",dcmColorModelfff,ž-
,,□f□ffff,-<,,,ffff,,,,,,,,',,,,,Œ',,□,fff,f□ffff,□',,,,,□□š□, f□ffff,□"
,',,□
ž,□f□ffffš□,□-<,,,ffff,Œ,ff□
□fff□,ffffff,□,,,,,,,,,,,,,"'",□*%",□□, D3DENUMRET_OK,* ,□,fff,š,,ž,□
-,ffff,□ž,ffff,-<,`,□f□ffffš□,□f□ff'<,BPPToDDBDš□,-,□'',,ff□
□fff□,□InitAppš□,GetDeviceCapsš□,Œ,□,.,.,.,ž",,□%\"",š,,□BPPToDDBD,b
its-per-pixel to DirectDraw bit-depth,-
,,□□BPPToDDBDš□, f□f,.,.,,□□BPPToDDBDfffš□□,ž□,.,.,□
-<,,,ffff,.,.,š',fff,□,Œ,□D3DDEVICEDESC□`\",',fff,'"',,□f□ffffš□,□fffff
f ffff□fff,.,,f□ffff,□ffff□fff,.,RGBfff,\"'\",□
//////////////////////////////////////
//
// enumDeviceFunc
// ž-%",D3Dffff,-',GUID,<%,,f□ffffš□□
// ffff,\"'\",□*lpContext,□',.,□
//
//////////////////////////////////////

static HRESULT
WINAPI enumDeviceFunc(LPGUID lpGuid, LPSTR lpDeviceDescription,
    LPSTR lpDeviceName, LPD3DDEVICEDESC lpHWDesc,
    LPD3DDEVICEDESC lpHELDesc, LPVOID lpContext)
{
    static BOOL hardware = FALSE; // Œ,šžffff,f□ffff,,,
    static BOOL mono = FALSE; // Œ,šžffff,ffffŒ,.,.
    LPD3DDEVICEDESC lpDesc;
    int *lpStartDriver = (int *)lpContext;

    // ,,ffff<□,' ,,,Œ',,□

    lpDesc = lpHWDesc->dcmColorModel ? lpHWDesc : lpHELDesc;

    // ∞∞
    //

```

```

if (!lpDesc->dwDeviceRenderBitDepth & BPPTToDDBD(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 (D3DENUMRET_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<,ffffff
,,ffffff,,Helworld.c,f□f,,,□3DŠ<,□',,••,,,□-,□^%,ffffff,□,,□-
,ž□,,,,,,,,,Š□,,,□-,,,,□
•    ffff,ff□f□f,□□
•    fffff□ff□f,□'

,,,Š□,□3DŠ<,ffffff,ff□f□f,□Œ,'",,,,,,,,□f□f,□□,□MySceneŠ□,□MySceneŠ□,□Œ,□,Š□Œ,,,,ŽŒ,,,
□3DŠ<,f□f,□',•-,,,,□□f□f,□□□,Ž□,,,,□
ffff,ff□f□f,□□
Direct3Dffff,ff□f□f,□ffff□fff,□Š%o
,Š,,,□□,,,□InitAppŠ□,□DirectDrawClipperffff□□,□Œ□DirectDrawClipperffff□,"",ffff□,
,,ffff□Œ,□-,fff□f,,,□CreateDevAndViewŠ□,□Œ,□,□
CreateDevAndViewŠ□,□—<ffff,"",ffff,Ž—
,,□IDirect3DRM::CreateDeviceFromClipperffff,,,Direct3DRMffff,□□,□,□IDirect3DRMDevic
eff□ffff,□ffff,•,,,□,„Ž“,„IDirect3DRMDevice::GetWidth,IDirect3DRMDevice::GetHeightfff
f,Œ,□,„Ž,,,□CreateDevAndViewŠ□,□ffff,•,□,□•,Ž,Ž,,Œ□IDirect3DRM::CreateViewportfff,
Œ,□,„IDirect3DRMViewportfff□ffff,Ž“,„□
Ž,CreateDevAndViewŠ□,□IDirect3DRMViewport::SetBackffff,,,ff□f□f,fff□ffff□ff□f,
□',,,,□f□f'<,SetRenderStateŠ□,Œ,□,,,□SetRenderStateŠ□,,,,□Ž,ffff□ffff□ff□f,□'□,
□-,□
////////////////////////////////////
//
// CreateDevAndView
// Ž',„D3Dffff,,fff,D3DRMffff,ff□f□f,□□,„□
//
////////////////////////////////////

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

    // Ž',„D3Dffff,—,□,,ffff,,D3DRMffff,□□,„□

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

    // fff□ff□f,Ž,D3DRMff□f□f,□□,□Œ,□,„,„□,
    // □',□•,□,,,,,□,,,,,□ffff,,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;
}

// ffffff,Ž"„,f□f□CE,□'□f□□ff□ffff,□•,
// □',□

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

```

```

fffff□ff□f,□'
Direct3D の

```

□

SetRenderState

SetRenderState

IDirect3DRMDevice::SetQuality

〇〇〇

IDirect3DRMDevice::SetDither

IDirect3DRMDevice::SetTextureQuality

の の

switch の IDirect3DRMDevice::SetShades

IDirect3DRM::SetDefaultTextureColors

IDirect3DRM::SetDefaultTextureShades

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

//

// SetRenderState

// の

//

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

BOOL

```

SetRenderState(void)
{
    HRESULT rval;

    //          ∞

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

    //  SetDither  ,
    //  fffff,Ž,D3DRMTEXTURE_NEARESTffffŒ,,,
    //  ,,,SetTextureQuality,€, ,
    //  €,ffff",,,,ffŒ,Œ,,,f%,f', ,
    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;
}

fffff f f
WinMainŠ , Ž, f f f , *%, , , , , RenderLoopŠ , Ğ, , RenderLoopŠ , , , , , ' , , -
, Ž , , ,
• IDirect3DRMFrame::Move f f f f , Ğ, , , Š '%o, , , , f f f , %o , " , " , — , , ,
• Direct3DRMViewport::Clear f f f f , Ğ, , , Ğ , , f f f f , " Ğ , f f f , ,
• IDirect3DRMViewport::Render f f f f , Ğ, , , Ğ , , f f f , f f f f , f f f f f , ,
• IDirect3DRMDevice::Update f f f f , Ğ, , , f f f f f , , f f f , f f f f , f f , ,

////////////////////////////////////
//
// RenderLoop
// f f f f , f f f , , Ž, f f f f , f f f f f , , f f f f f , , , , ,
//
////////////////////////////////////

static BOOL
RenderLoop()
{
    HRESULT rval;

    // Ğ , , f f , Š , , ,

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

    // f f f f , f f f , , ,

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

    // f f f , f f f f , f f f f f , , ,

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

```

```

// fffff,□□,□

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

f□f,□□
3DŠ<,ffffff□ffff,“□3Dffff,ff□f□f,□□□ffffff□ff□f,□',□,Š—
,,,□Helworld.c,□,,3DŠ<,ffffff,ff□f□Œ,“,,,,,Š□Œ,Œ,□,□
• MySceneŠ□
• MakeMyFramesŠ□
• MakeMyLightsŠ□
• SetMyPositionsŠ□
• MakeMyMeshŠ□
• MakeMyWrapŠ□
• AddMyTextureŠ□

MySceneŠ□
Helworld.c,MySceneŠ□,□DirectX
SDK,,,,Direct3Dffff.ffffff,,,,□BuildSceneŠ□,“,“,□ffff□fff,ffffff.ffff,□—Œ%,,,,•Ž,□
—,□,,,,Š□“,□,□□
MySceneŠ□,□□□,,,f□f,Š“□,□',□f□f'<,Š□Œ,Œ,□,□,,,,Š□,~%,Ž,□
• MakeMyFrames
• MakeMyLights
• SetMyPositions
• MakeMyMesh
• MakeMyWrap
• AddMyTexture

,,,Š□,ffffff□ffffff,ffffff,Š—
,,,□MySceneŠ□,IDirect3DRMFrame::AddVisualfff,Œ,□,□ffffff,3DŠ<,w
orldff□f,%o,,□,,Œ,□□,,
Release の

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

// MyScene
// ff f ŒŒ ffff fffff, ,Š ,Œ, , Š—,, ,,,
// fff ffff,%o•,,
//
////////////////////////////////////

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);

    // fffff,00,,*-,,,000,,,,0E,<,-,00,,0000
    // ,,CreateMaterial,SetMaterial,E,0,0

    // ,,ffffff0ffffff,00,,,,,0f0ff0ff0f,'%,,0

    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;
}

MakeMyFramesS0
MySceneS0,MakeMyFramesS0,E,0,0Helworld.c,-
,,,,ffffff0Efff0f,f0ff0ff0f,00,,0MakeMyFramesS0,0IDirect3DRM::Create
Frameffff,E,0,,0,-,Ž0,,0
////////////////////////////////////
//
// MakeMyFrames
// f0f,Ž-,,ff0f,00,,0
//
////////////////////////////////////

void MakeMyFrames(LPDIRECT3DRMFRAME lpScene, LPDIRECT3DRMFRAME
lpCamera,

```

```

        LPDIRECT3DRMFRAME * lpPlpLightFrame1,
        LPDIRECT3DRMFRAME * lpPlpWorld_frame)
{
    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, lpScene, lpPlpLightFrame1);
    lpD3DRM->lpVtbl->CreateFrame(lpD3DRM, lpScene, lpPlpWorld_frame);
}

```

```

MakeMyLightsŠ
MySceneŠ, MakeMyLightsŠ, Ą, , Helworld.c, -
,,, fffffffĄ, fffffffĄ, , , MakeMyLightsŠ, IDirect3DRM::CreateLigh
tRGB, IDirect3DRMFrame::AddLightffff, Ą, , Ą•Ą, Ź, -, , ĄĄ, , , fĄf, '%
,, , , Ą"ˆ, fffffffĄ, , , ĄfĄ`', '%,, , fffffffĄ, , , fĄf`', Š•, , , ,
////////////////////////////////////
//
// MakeMyLights
// fĄf, Ź-, , ĄĄ, , ,
//
////////////////////////////////////

```

```

void MakeMyLights(LPDIRECT3DRMFRAME lpScene,
LPDIRECT3DRMFRAME lpCamera,

    LPDIRECT3DRMFRAME lpLightFrame1,

    LPDIRECT3DRMLIGHT * lpPlpLight1, LPDIRECT3DRMLIGHT *
    lpPlpLight2)
{
    lpD3DRM->lpVtbl->CreateLightRGB(lpD3DRM,
    D3DRMLIGHT_DIRECTIONAL,
    D3DVAL(0.9), D3DVAL(0.9), D3DVAL(0.9), lpPlpLight1);

    lpLightFrame1->lpVtbl->AddLight(lpLightFrame1, *lpPlpLight1);

    lpD3DRM->lpVtbl->CreateLightRGB(lpD3DRM,
    D3DRMLIGHT_AMBIENT,
    D3DVAL(0.1), D3DVAL(0.1), D3DVAL(0.1), lpPlpLight2);

    lpScene->lpVtbl->AddLight(lpScene, *lpPlpLight2);
}

```

```

SetMyPositionsŠ
MySceneŠ ,SetMyPositionsŠ ,Ą, , Helworld.c,Ź,ff f,ˆ,Ą,, ',, SetMy
PositionsŠ , , —
, IDirect3DRMFrame::SetPosition,, , IDirect3DRMFrame::SetOrientationffff

```



```

,Œ, ,,,,Ž ,, IDirect3DRMFrame::SetRotationfff, <‘,%o
,,,fff f,%o“, ’,,
////////////////////////////////////
//
// SetMyPositions
//
//
//
//
////////////////////////////////////

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));
}

```

[illegible]

```
MakeMyWrapŠ□  
MySceneŠ□, MakeMyWrapŠ□, €□, , fffff□•, □□, □MakeMyMeshŠ□, f□f□, < \, “—  
□, □MakeMyWrapŠ□, □< \, Š, •€< Š□, Ž“, , , , IDirect3DRMMeshBuilder::GetBoxffff  
€, □, □, , •€< Š□, fff, IDirect3DRM::CreateWrapffff, €, □, Ž, Ž—  
□, □IDirect3DRM::CreateWrapffff, %‘, fffff□fff, □□, □IDirect3DRMWrapffff□ff  
ff, Ž“, □ffffff□•, < \, “—, , , , IDirect3DRMWrap::Applyffff, €, □, □, □  
/////////////////////////////////////  
//  
// MakeMyWrap  
// fff, □□, □ffffff, “—, □  
//  
////////////////////////////////////
```

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

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

    maxy = box.max.y;
```

## Chapter 5 Part A: Direct3D Overviews 67

```

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, E, , , , fffff, f, < `,, Š~*,, , AddMyTextureŠ,
IDirect3DRM::LoadTextureffff,-
, Hello.ppm,, -`, fffff, f, , , , IDirect3DRMMeshBuilder::SetTexture,
E, , , fffff, < `,, , , Hello.ppm, 256256, 256ffff, , ,
$
////////////////////////////////////
//
// AddMyTexture
// fff, , , , fffff, "-", , ,
//
////////////////////////////////////

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

    // fffff16Š, ff, *, -, , , ,
    // IDirect3DRMTexture::SetShades, E, , ,

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

}

- -
Helworld.c, WM_DESTROY fff, Š, Š, , , , , , RenderLoopŠ, E, , , %", Š", , , , Cl
eanUpŠ, E, , ,
////////////////////////////////////
//
// Cleanup
// , , , , D3DRM fffff, %*, bQuit fff, fff, , ,
//
////////////////////////////////////

```

```
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;
}

'□f□f,š-
'□f□f,,,,
,,fffff,,□Microsoft,'fff3D API,,, '□f□f,,,,□-
,,□Direct3D,'□f□f,□f□f,,, '\,ff□fff□fff,ffffffffff□f □□□□□□
□□Microsoft Windows の
□
Direct3D の
の
の
□□□□□□□□
3D の
□□□□□□□□
3D の
のDirect3D のDirect3D のDirect3D
Direct3D
の
: Direct3D ののの SDK
Direct3D の
のDirect3D ののの Direct3D の
: Direct3D の
Direct3D の □□□□□□□□□□の API
Direct3D の □□□□□□□□□□ の
の Direct3D
のの8 の COM
COM
```

IDirect3D

COM

IDirect3DDevice

IDirect3DTexture

DirectDraw

IDirect3DMaterial

IDirect3DLight

IDirect3DViewport

*IDirect3DDevice*

4 4 の

□□ ***ffff***  
**IDirect3DExecuteBuffer*****fff***□***ffff***

”“***f***□***f,fff,ffffff•–,ŽŽ***

*ffff*  
COM  
の ∞ の

- *COM*
- COM の の の COM の
- の

,, Direct3D,Šff  
ffff,‘ ,,ŒŽ,Ž,  
,,

COMfff ffff fffff  
“  
ffff

ffff fffff

ŒŒ

ff f f Textureffffff

Materialffffff

—

Lightffffff

Ž ffff Viewportffffff

Direct3DffffffŒ  
,,ffff,, Direct3  
D,ffffffŒ,fff  
ffff,,, –  
,, ffff fff, Ž  
,’,,, , Direct3  
Dffffff,fff ff  
ff, ,,,,,,  
Direct3Dfff fff  
f ffffff

Deviceffffff

Execute-bufferffffff

IDirect3Dfff ffff

IDirect3Dfff ffff, DirectDrawffffff  
,,fff ffff,, IDirect3Dfff ffff,  
,,, Ž,,,IDirectDraw2::QueryInterface  
fff,Œ, ,, ,

lpDirectDraw->QueryInterface(  
IID\_IDirect3D, //  
IDirect3Dfff ffff,ID  
lpD3D); //  
Direct3Dffffff,ffff

IDirect3Dfff ffff,,,Ž ,,ffffff,  
ff f f ŒŒ ffff ,,ffff,fff  
,Š,,, IDirect3D,ffff, ‘,ffffff,  
,, Direct3Dfff,Œ,,,,Ž—  
,,,,,,

IDirect3DDevicefff ffff

IDirect3DDevicefff ffff, DirectDr  
aw

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



```

lpDirectDraw->CreateSurface(
    lpDDSurfDesc, // DDSURFACEDESC の
    lpFrontBuffer, // DIRECTDRAWSURFACE の
    pUnkOuter); // NULL
lpFrontBuffer->GetAttachedSurface(
    &ddscaps, // DDSCAPS “,,ffff
    &lpBackBuffer); // DIRECTDRAWSURFACE “,,ffff
lpBackBuffer->QueryInterface(
    GUIDforID3DDevice, // IDirect3DDevicefff ffff,ID
    lpD3DDevice); // DIRECT3DDEVICEffffff,,ffff

```

```

ffffff,,,,IDirectDrawSurface::QueryInterfaceffff,☒,☐,Ž’,,, ,fff☐f,☐I
Direct3DDevicefff☐ffff,Ž,☐ff☐ff☐fff,—
^,Ž•Ž☐GUID☐,,,☐,,GUID,☐IDirect3D::EnumDevicesffff,☒,☐,,,,Ž“,,,,,,,
fff      IDirect3D::EnumDevices のの D3DENUMDEVICESCALLBACK

```

```

ffŠ ,☒, ,,, ,,
GUID,☐,,
Direct3D,ffff f
fffff, fff fff
f fff,‘ , ,,, ‘
, Ž ,Ž ffff
,ff f f,fff,Ž,,
, ,, fffff,ffff
f,fff,Š,,, ,,,ff
f’,Ž,fffff,fffff,
,ffff, ‘,ffff,,ff
ff,—
•,Ž,,, ,Š‘ ‘,,,
,,, ffffff, ‘
,

```

```

Ž☐,,,,☐
IDirect3DDevicefff☐ffff,ff のの

```

```

Direct3D の IDirect3DDevice::CreateMatrix
IDirect3DDevice::SetMatrix
ffff,Ž ffff,—,,,

```

*IDirect3DTexture*fff ffff  
 fffff, •,ffff,⌢E,•,,,,, ⌢E,•,, •E,,,  
 ,,, ffff, ,E  
 —“,“ ,,, •E, ,,, fffff,ffffff ffff  
 f,—  
 ,,Ž,,,,,, , , •E,,,,,,, RGBff ff  
 f,—,ffff fff,, 8 24 32fff,ffffff,Ž—  
 ,,,,,, ffff fff ff fff,, 8fff,ffff  
 f,,Ž

,”□  
*IDirect3DTexture*fff□ffff,□*DirectDrawSurface*  
*IDirect3DTexture*  
*IDirectDrawSurface2::QueryInterface* IID\_ *IDirect3DTexture*  
*Direct3D の DirectDraw の* **Direct3D**

の*IDirect3DTexture* □□□□□□□□□□—  
 ,□*IDirect3DTexture::GetHandle*,,, *IDirect3DTexture::Load*fff,—  
 ,,ffffff,f□f,,•-,Ž,,,,□  
 lpDDS->*QueryInterface*(IID\_ *IDirect3DTexture*,  
 lpD3DTexture); // *DIRECT3DTEXTURE*ffffff,,ffff  
 lpD3DTexture->*GetHandle*(  
 lpD3DDevice, // *DIRECT3DDEVICE*  
 lpTexture); // *D3DTEXTUREHANDLE* の  
 lpD3DTexture->*Load*(  
 lpD3DTexture); // *DIRECT3DTEXTURE* の

¥ののののの  
 ,ffffff,Ž□ffff,—,,□ff□f□f,Zffff,,Ž,,□ffffff,f□ のの  
*IDirect3DTexture*

の*Direct3D* のの

- 
- 
- 
-

①①

のD3DRENDERSTATETYPE  
D3DRENDERSTATE\_WRAPV

D3DRENDERSTATE\_WRAPU

$$\begin{array}{ccccc} & & & & \mathcal{O} \\ & & & \mathcal{O} & \\ & v & & & \\ \mathbf{u} & & \mathbf{v} & & \mathcal{O} \end{array}$$

○○○

 $\mathcal{U}$ 

•  $\square\square\square f f\square f f f, f f f, , , , , f f f f\square f f f f f\square f\square f, , \square u, , , v\square\cdot, \tilde{z}', , , \cdot-, \square f f f f f, -$   
 $\mathbb{E}, f f f f f, , , \square, , \square\square\square-\mathbb{E}, u, , , v\square\cdot', 1.0^\wedge\square, , , \square''(0.1, \ 0.1), (0.9, \ 0, 9)$   
 $, \mathbb{E}, \square', ' \square, \square''(0.5, \ 0.5), ', \% , \square$   
 •

D3DRENDERSTATE\_WRAPU,,,D3DRENDERSTATE\_WRAPV,,,,,fff,,,□□□fffff,%Ž1.0,□, ,%„Œ„,□1.0^□,fffff□•,□fff,,,,,„“—Œ„,□fffff□•Š,□'<— ,fffff□fff,^,□D3DRENDERSTATE\_WRAPU,fff,,,,,□(0.1, 0.1),(0.9, 0.9),,,,□,'□,□□0, 0.5□,%„,□

- D3DRENDERSTATE\_WRAPU,D3DRENDERSTATE\_WRAPVfff,— ,fff,,,,,□fffff,%Š□□□fff□□,,,□ffff,•,,,,□1.0^□,fffff□•,—Œ„,□(0.1, 0.1),(0.9, 0.9) ,,,□,'□,□□(0, 0),%„,□

$$\begin{aligned} & -\mathbb{E}^{\wedge},,,,,fffff\Box,\text{---}\mathbb{E}^{\wedge},\check{Z},,,,,,\Box,,,,,\text{“}\Box,\Box,,,,,\Box \\ & ^{\wedge},,,,,fffff\Box fff,\Box fffff,\mathbb{E}\Box,-,\text{“},^{\wedge}',,,,,,\Box\%'\Box,fff,,,,,fffff\Box fff,fff,,\Box,,,,-,fffff,\text{”}\cdot^{\wedge}\Box,\text{---} \\ & ,,,\Box\Box,\Box fffff\Box fff,\Box',,,\Box \\ & fffff,.,.,\Box\Box fffffffffff:\text{Direct3D}\cdot\check{Z}f\Box ffffff\Box,\Box|\text{Direct3D}\text{RMWrap}fff\Box ffff\Box,\check{Z}\Box,,,,,\Box \\ & fff \end{aligned}$$

④④④④

D3DRENDERSTATETYPE

D3DRENDERSTATE\_TEXTUREMAG  
D3DRENDERSTATE\_TEXTUREMIN

### D3DRENDERSTATE TEXTUREMAPBLEND

の

)))

D3DTEXTUREBLEND

<u>OD3DRENDERSTATE SRCBLEND</u>	D3DRENDERSTATE DESTBLEND
---------------------------------	--------------------------

D3DBLEND

D3DTEXTUREFILTER の  
D3DPRIMCAPS の dwTextureFilterCaps

DirectDraw の

D3DRENDERSTATE\_BLENDENABLE D3DRENDERSTATETYPE の

の DirectDraw

の

DirectDraw の

**IDirect3DMaterial** の

IDirect3DMaterial の IDirect3D::CreateMaterial の

IDirect3DMaterial の IDirect3DMaterial::SetMaterial

IDirect3DMaterial::GetHandle の

lpDirect3D->CreateMaterial(

lpDirect3DMaterial, //

pUnkOuter); // NULL

lpDirect3DMaterial->SetMaterial(

lpD3DMat); // D3DMATERIAL の

lpDirect3DMaterial->GetHandle(

lpD3DDevice, // DIRECT3DDEVICE の

lpD3DMat); // D3DMATERIAL の

の

の

の

IDirect3DMaterial の

IDirect3DLight

IDirect3DLight IDirect3D::CreateLight

の IDirect3DLight の IDirect3DLight::SetLight

**lpDirect3D->CreateLight(**

lplpDirect3DLight, //

```
pUnkOuter);    // NULL
```

**lpDirect3DLight->SetLight(****lpLight); // D3DLIGHT** ④

```

IDirect3DLight ffff, ž-, ,E, ž", □', □, , , , , , □
IDirect3DViewport fff□ffff
IDirect3DViewport fff□ffff, □ IDirect3D::CreateViewport ffff, E, □, , □□, , □ž,
-, □ IDirect3DViewport fff□ffff, □□*- , ž, , , , □, □, -,
, □ IDirect3DDevice::AddViewport ffff, , , ff□f□f, '%
, , , , □ IDirect3DViewport::SetViewport □ IDirect3DViewport::SetBackground
□ IDirect3DViewport::AddLight ffff, -, , ff□f□f, fffffff, *- , □-, , , , □
lpDirect3D->CreateViewport(
    lpDirect3DViewport, // □, , ff□f□f, , ffff
    pUnkOuter);        // NULL
lpD3DDevice->AddViewport(
    lpD3DViewport)      // ff□f□f, ffff, ffff, ,
lpD3DViewport->SetViewport(
    lpData);            // のの

// D3DVIEWPORT の

lpD3DViewport->SetBackground(
    lpMat);             // の D3DMATERIALHANDLE の

lpD3DViewport->AddLight(
    lpD3DLight);        // の

```

¥①

IDirect3DViewport

## IDirect3DDeviceExecuteBuffer

ののŽ ffff, -, Ž ffff ,Ž ,,,,

```
IDirect3DExecuteBufferfff ffff,    ,,, IDirect3DDevice::CreateExecuteBu
fferfffff,0,  ,  ,
```

lpD3DDevice->CreateExecuteBuffer(

```
lpDesc,    // DIRECT3DEXECUTEBUFFERDESC “,ffff
```

```
lpDirect3DExecuteBuffer, // Direct3DExecuteBufferfffff,,
```

// ffff,Ž,Ž,,,,ffff

pUnkOuter); // NULL

Ž□ffff,ffff□fff,Š,,,,□Ž□,ffff—  
^,Š•,,,,□IDirect3DDevice::CreateExecuteBuffer の

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

lpD3DExBuf->Lock(  
lpDesc);. // **DIRECT3DEXECUTEBUFFERDESC** の  
// .  
// . **Store contents through the supplied address**  
// .  
**lpD3DExBuf->Unlock();**  
lpD3DExBuf->SetExecuteData(  
lpData); // D3DEXECUTEDATA の

のの IDirect3DExecuteBuffer::SetExecuteData ののDirect3D  
IDirect3DExecuteBuffer::Lock

IDirect3DExecuteBuffer  
の  
ff f f, —  
,,,, ,,,,ff f f,,,,,ŠŒŒ,•Ž,,,,,Ž,,,,,  
§  
ffffff,ffffff,ffffff, ,,^ ,ffff,Š~•,,,,,, Ž, ,Ž,,,Šf f, fff, “,ffff,•  
Ž,,, ,, ,',ffff,•Ž,,,,, ,,ffffff,f f,,, ,ffffff,f f,,, ,ffffff,  
“,Š~,ffff,ffff,Ž“,,,,,,  
§  
fff ffff,ffff fffff,ffffff, ,,,QueryInterfaceffff,Œ, , fffff Œ  
Œ ff f f,ffffff, IDirect3Dfff ffff,ffff,Œ, ,, ,,,,,, Ž fff  
f, —  
, IDirect3DDevicefff ffff,,, ,,, fff ffff fffffff Direct3D  
object ,, ,,,,,,ffff,ff f f ŒŒ fffff,ffffff,fff,•Ž,,, Ž ff  
ff,ffffff,ffffff,Š,,,  
f f,Š—  
,,,’ f f ffff fff, fffffff,Ž ,,‘,IDirect3DDevice::BeginSceneffff,Œ,  
,,,,,, , fffffff,Š—,,,,, IDirect3DDevice::EndSceneffff,Œ, ,•—  
,, ffff fff, Œ,3Df ffff “ ,,,,, ,,,ffff, ,,Ž—,,,,,,

```
f f,"•%ffffff ŽŠŒ ' " , ' , , " , "%o,fff fff,,,,ff f,•%
,,,,Ž—
,,, IDirect3DDevice::BeginSceneffff,f f,Ž,,, ,,,,ff f,ŠŽ, Ž, IDirect3DDevice::EndSceneffff,f f,ff f, —," , , , , ff f,•%o
,Ž,,,Ž ffff—
, 1“,IDirect3DDevice::BeginScene,IDirect3DDevice::EndScene,Œ, ,Š,,,
,,,,,,
,,fffff,^%o,ffff, , , —, , ,
• %o— <
• 2D,3D,‘Œ —
• f fŠ—fff ff
```

```
%o— <
3Dfffff□f,” fff,,□□—,Zffff
IDirect3DDevice::BeginScene
IDirect3DDevice::EndScene
```

```
のののの
の
```

```
のののののの
1 のIDirect3DDevice::BeginScene IDirect3DDevice::EndScene の
,,,,,,□,,,ff□f,□□',□•□%o,IDirect3DDevice::BeginScene
IDirect3DDevice::EndScene の
```

```
のの
IDirect3DDevice::BeginScene IDirect3DDevice::EndScene
```

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

**2D 3D の**

```
IDirect3DDevice::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 IDirect3DDevice::BeginScene
Direct3DDevice::EndScene
```





- *ReleaseScene*
- *OverrideDefaults*

```
,,, ffff,SetMouseCallbackŠ ,SetKeyboardCallbackŠ ,Œ, , fff,f f f,,,
“—,Ž“,,,
```

Step 1: Š‰,ŠŽ

```
D3dmain.cpp,WinMainŠ□,□,□□,fff,□f□ff,‘<,,,AppInitŠ□,Œ,□,□ffff□ff
f□fffff,□□,□fffff,ŠŽ,•—,,,,fffff,□Š‰
,,□,,□WinMainŠ□,D3dmain.cpp,fff□f□fff,fffff,□f□ff‘<,RenderLoopŠ
□,CleanupAndPostQuit AppInit ののののののの
```

```
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
```

```
WinMainŠ “,,,ffff fff fffff,“,,, CreateD3DAppŠ „,,, —
Œ,fffff, -systemmemory,-emulation,,, -systemmemoryfffff, ffff
—,—,,, -
emulationfffff,Ž’,,, ffff fff,DirectDraw,Direct3D,f ffff fffff f,
Ž—,,,
```

```
CreateD3DAppŠ□,□ffff,□□,,,,D3DAppAddTextureŠ□,Œ,□,□,,□D3DA
ppAddTextureŠ□,□f□f□fffff,f□fff,fffff,ffff□fff“, „, f□ffff,ff
f□fff,Ž,,,□□,□□Š□‘<,fffff□f□fff,fff□fff“, „, f□f□ffff,□“
,fffff□f□fff,f□f,,Œ,”Š,,,□,2’Š,ffff,,,□ffff,□fff□fff,“,,,,,ffff
f□fff,^□,,□ff□fff,,,,,□ffff□f□f,,□IDirect3DTexturefff□ffff,Ž“,,,
,,IDirectDrawSurface::QueryInterfacefff,Œ,□,□IDirect3DTexture::Load
のIDirect3DTexture::GetHandle
```

```
のCreateD3DApp の DirectDraw
Direct3Dfffff,□□,,□fff□f□f,,□D3DAppCreateFromHWND
D3DAppCreateFromHWND D3dapp.c
D3dcalls.c Texture.c Ddcalls.c
```

```
D3DAppCreateFromHWND DirectDrawEnumerate
DirectDrawCreate DirectDraw のの
IDirectDraw::EnumDisplayModes
```

IDirectDraw の IDirectDraw2 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

GUID

の

の

GUID

D3DAppCreateFromHWND の の の の IDirectDraw::CreateClipper

DirectDrawClipper ffffff, □□, □ IDirectDrawClipper::SetHWND ffffff, ffffff  
ff, ffffff, S~, □ 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

CreateD3DApp

の

の

Step 3: の

の

D3DAppCreateFromHWND の Step 5: の

Direct3D

の

D3DAppCreateFromHWND TRUE

D3DAppCreateFromHWND の の Direct3D

DirectDraw

FALSE

Step 3: の

D3DAppCreateFromHWND の 3 AfterDeviceCreated  
D3dmain.cpp  
AfterDeviceCreated 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

Oct1.c InitView の  
InitView の

InitView IDirect3D::CreateMaterial  
**IDirect3DMaterial::SetMaterial**  
IDirect3DMaterial::GetHandle IDirect3DViewport::SetBackground

**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**  
D3DEXECUTEBUFFERDESC の**IDirect3DDevice::CreateExecuteBuffer**  
IDirect3DExecuteBuffer::Lock

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

## InitView

## IDirect3DExecuteBuffer::Unlock

## IDirect3DExecuteBuffer::SetExecuteData

**IDirect3DDevice::BeginScene**   **IDirect3DDevice::Execute**

IDirect3DDevice::EndScene

IDirect3DExecuteBuffer::Release

InitView    IDirect3D::CreateMaterial    **D3DMATERIAL**    の

IDirect3DMaterial::SetMaterial

IDirect3DMaterial::GetHandle の D3DLIGHTSTATETYPE の

D3DLIGHTSTATE MATERIAL

## InitView の D3DVERTEX の D3DVALUE

D3DVALP X

D3DRMVectorNormalize

⓪InitView ⓪⓪⓪⓪

InitView      Oct1.c のD3DLIGHT

IDirect3D::CreateLight    IDirect3DLight::SetLight

IDirect3DViewport::AddLight

Step 5:  $\mathcal{O}(\mathcal{O})$

D3dcalls.c D3DAppISetRenderState の

D3DAppCreateFromHWND      D3dapp.c      **D3DAppISetRenderState**

D3DAppISetRenderState の実装

## D3DAppISetRenderState

D3DAppISetRenderState D3DEXECUTEBUFFERDESC

D3DEXECUTEDATA ④

## 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 d3dappi.lpD3DDevice
Direct3DDevice の debDesc      D3DEXECUTEBUFFERDESC の
lpData

      D3DAppISetRenderState          の
D3DAppISetRenderState      OP_STATE_DATA          の
      PUTD3DINSTRUCTION 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 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(lpBuffer);
```

```
D3DAppISetRenderState
```

```
IDirect3DExecuteBuffer::Unlock
```

```
    IDirect3DExecuteBuffer::SetExecuteData
```

```
    IDirect3DDevice::BeginScene  IDirect3DDevice::Execute
```

```
IDirect3DDevice::EndScene
```

```
LastError = lpD3DExCmdBuf->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;
```

```
}
```