

DirectX 11 CheatSheet

Version 1.0 © 2011-09-18 Adam Sawicki, www.asawicki.info

General

IDXGISwapChain

```
typedef struct DXGI_SWAP_CHAIN_DESC {
    DXGI_MODE_DESC    BufferDesc;
    DXGI_SAMPLE_DESC  SampleDesc;
    DXGI_USAGE        BufferUsage;
    UINT              BufferCount;
    HWND              OutputWindow;
    BOOL              Windowed;
    DXGI_SWAP_EFFECT  SwapEffect;
    UINT              Flags;
} DXGI_SWAP_CHAIN_DESC;
```

```
typedef struct DXGI_SAMPLE_DESC {
    UINT Count;
    UINT Quality;
} DXGI_SAMPLE_DESC, *LPDXGI_SAMPLE_DESC;
```

```
typedef enum DXGI_SWAP_EFFECT {
    DXGI_SWAP_EFFECT_DISCARD    = 0,
    DXGI_SWAP_EFFECT_SEQUENTIAL = 1
} DXGI_SWAP_EFFECT, *LPDXGI_SWAP_EFFECT;
```

```
typedef enum DXGI_SWAP_CHAIN_FLAG {
    DXGI_SWAP_CHAIN_FLAG_NONPREROTATED    = 1,
    DXGI_SWAP_CHAIN_FLAG_ALLOW_MODE_SWITCH = 2,
    DXGI_SWAP_CHAIN_FLAG_GDI_COMPATIBLE    = 4
} DXGI_SWAP_CHAIN_FLAG, *LPDXGI_SWAP_CHAIN_FLAG;
```

```
typedef struct DXGI_MODE_DESC {
    UINT          Width;
    UINT          Height;
    DXGI_RATIONAL RefreshRate;
    DXGI_FORMAT   Format;
    DXGI_MODE_SCANLINE_ORDER ScanlineOrdering;
    DXGI_MODE_SCALING Scaling;
} DXGI_MODE_DESC, *LPDXGI_MODE_DESC;
```

```
typedef struct DXGI_RATIONAL {
    UINT Numerator;
    UINT Denominator;
} DXGI_RATIONAL, *LPDXGI_RATIONAL;
```

```
typedef enum DXGI_MODE_SCALING {
    DXGI_MODE_SCALING_UNSPECIFIED = 0,
    DXGI_MODE_SCALING_CENTERED    = 1,
    DXGI_MODE_SCALING_STRETCHED   = 2
} DXGI_MODE_SCALING, *LPDXGI_MODE_SCALING;
```

```
typedef enum D3D_FEATURE_LEVEL {
    D3D_FEATURE_LEVEL_9_1    = 0x9100,
    D3D_FEATURE_LEVEL_9_2    = 0x9200,
    D3D_FEATURE_LEVEL_9_3    = 0x9300,
    D3D_FEATURE_LEVEL_10_0   = 0xa000,
    D3D_FEATURE_LEVEL_10_1   = 0xa100,
    D3D_FEATURE_LEVEL_11_0   = 0xb000
} D3D_FEATURE_LEVEL;
```

ID3D11InputLayout

```
typedef struct D3D11_INPUT_ELEMENT_DESC {
    LPCSTR          SemanticName;
    UINT            SemanticIndex;
    DXGI_FORMAT     Format;
    UINT            InputSlot;
    UINT            AlignedByteOffset; // D3D11_APPEND_ALIGNED_ELEMENT
    D3D11_INPUT_CLASSIFICATION InputSlotClass;
    UINT            InstanceDataStepRate;
} D3D11_INPUT_ELEMENT_DESC;
```

```
typedef enum D3D11_INPUT_CLASSIFICATION {
    D3D11_INPUT_PER_VERTEX_DATA = 0,
    D3D11_INPUT_PER_INSTANCE_DATA = 1
} D3D11_INPUT_CLASSIFICATION;
```

MISC

IDXGIObject
IDXGIDeviceSubObject
ID3D11DeviceChild
ID3D11Resource
ID3D11View

ID3D11Device
ID3D11DeviceContext
ID3D11CommandList

typedef ID3D10Blob ID3DBlob;

```
typedef struct D3D11_VIEWPORT {
    FLOAT TopLeftX;
    FLOAT TopLeftY;
    FLOAT Width;
    FLOAT Height;
    FLOAT MinDepth;
    FLOAT MaxDepth;
} D3D11_VIEWPORT;
```

```
typedef RECT D3D11_RECT;

typedef struct D3D11_BOX {
    UINT left;
    UINT top;
    UINT front;
    UINT right;
    UINT bottom;
    UINT back;
} D3D11_BOX;
```

```
typedef enum D3D11_PRIMITIVE {
    D3D11_PRIMITIVE_UNDEFINED = 0,
    D3D11_PRIMITIVE_POINT = 1,
    D3D11_PRIMITIVE_LINE = 2,
    D3D11_PRIMITIVE_TRIANGLE = 3,
    D3D11_PRIMITIVE_LINE_ADJ = 6,
    D3D11_PRIMITIVE_TRIANGLE_ADJ = 7,
    D3D11_PRIMITIVE_1_CONTROL_POINT_PATCH = 8,
    ...
    D3D11_PRIMITIVE_32_CONTROL_POINT_PATCH = 39
} D3D11_PRIMITIVE;
```

```
typedef enum D3D11_PRIMITIVE_TOPOLOGY {
    D3D11_PRIMITIVE_TOPOLOGY_UNDEFINED = 0,
    D3D11_PRIMITIVE_TOPOLOGY_POINTLIST = 1,
    D3D11_PRIMITIVE_TOPOLOGY_LINELIST = 2,
    D3D11_PRIMITIVE_TOPOLOGY_LINESTRIP = 3,
    D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST = 4,
    D3D11_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP = 5,
    D3D11_PRIMITIVE_TOPOLOGY_LINELIST_ADJ = 10,
    D3D11_PRIMITIVE_TOPOLOGY_LINESTRIP_ADJ = 11,
    D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST_ADJ = 12,
    D3D11_PRIMITIVE_TOPOLOGY_TRIANGLESTRIP_ADJ = 13,
    D3D11_PRIMITIVE_TOPOLOGY_1_CONTROL_POINT_PATCHLIST = 33,
    ...
    D3D11_PRIMITIVE_TOPOLOGY_32_CONTROL_POINT_PATCHLIST = 64
} D3D11_PRIMITIVE_TOPOLOGY;
```

```
typedef enum D3D11_CLEAR_FLAG {
    D3D11_CLEAR_DEPTH = 0x1L,
    D3D11_CLEAR_STENCIL = 0x2L
} D3D11_CLEAR_FLAG;
```

```
typedef enum D3D11_COLOR_WRITE_ENABLE {
    D3D11_COLOR_WRITE_ENABLE_RED = 1,
    D3D11_COLOR_WRITE_ENABLE_GREEN = 2,
    D3D11_COLOR_WRITE_ENABLE_BLUE = 4,
    D3D11_COLOR_WRITE_ENABLE_ALPHA = 8,
} D3D11_COLOR_WRITE_ENABLE;
```

```
DXGI_RESOURCE_PRIORITY_MINIMUM
DXGI_RESOURCE_PRIORITY_LOW
DXGI_RESOURCE_PRIORITY_NORMAL
DXGI_RESOURCE_PRIORITY_HIGH
DXGI_RESOURCE_PRIORITY_MAXIMUM
```

```
typedef enum D3D11_TEXTURECUBE_FACE {
    D3D11_TEXTURECUBE_FACE_POSITIVE_X = 0,
    D3D11_TEXTURECUBE_FACE_NEGATIVE_X = 1,
    D3D11_TEXTURECUBE_FACE_POSITIVE_Y = 2,
    D3D11_TEXTURECUBE_FACE_NEGATIVE_Y = 3,
    D3D11_TEXTURECUBE_FACE_POSITIVE_Z = 4,
    D3D11_TEXTURECUBE_FACE_NEGATIVE_Z = 5
} D3D11_TEXTURECUBE_FACE;
```

Resources

```
typedef enum D3D11_USAGE {
    D3D11_USAGE_DEFAULT      = 0,
    D3D11_USAGE_IMMUTABLE   = 1,
    D3D11_USAGE_DYNAMIC      = 2,
    D3D11_USAGE_STAGING     = 3
} D3D11_USAGE;
```

```
typedef enum D3D11_BIND_FLAG {
    D3D11_BIND_VERTEX_BUFFER      = 0x1L,
    D3D11_BIND_INDEX_BUFFER       = 0x2L,
    D3D11_BIND_CONSTANT_BUFFER    = 0x4L,
    D3D11_BIND_SHADER_RESOURCE    = 0x8L,
    D3D11_BIND_STREAM_OUTPUT      = 0x10L,
    D3D11_BIND_RENDER_TARGET      = 0x20L,
    D3D11_BIND_DEPTH_STENCIL      = 0x40L,
    D3D11_BIND_UNORDERED_ACCESS   = 0x80L
} D3D11_BIND_FLAG;
```

```
typedef enum D3D11_CPU_ACCESS_FLAG {
    D3D11_CPU_ACCESS_WRITE = 0x10000L,
    D3D11_CPU_ACCESS_READ  = 0x20000L
} D3D11_CPU_ACCESS_FLAG;
```

```
typedef enum D3D11_RESOURCE_MISC_FLAG {
    D3D11_RESOURCE_MISC_GENERATE_MIPS          = 0x1L,
    D3D11_RESOURCE_MISC_SHARED                = 0x2L,
    D3D11_RESOURCE_MISC_TEXTURECUBE         = 0x4L,
    D3D11_RESOURCE_MISC_DRAWINDIRECT_ARGS   = 0x10L,
    D3D11_RESOURCE_MISC_BUFFER_ALLOW_RAW_VIEWS = 0x20L,
    D3D11_RESOURCE_MISC_BUFFER_STRUCTURED   = 0x40L,
    D3D11_RESOURCE_MISC_RESOURCE_CLAMP     = 0x80L,
    D3D11_RESOURCE_MISC_SHARED_KEYEDMUTEX  = 0x100L,
    D3D11_RESOURCE_MISC_GDI_COMPATIBLE     = 0x200L
} D3D11_RESOURCE_MISC_FLAG;
```

```
typedef struct D3D11_SUBRESOURCE_DATA {
    const void *pSysMem;
    UINT        SysMemPitch;
    UINT        SysMemSlicePitch;
} D3D11_SUBRESOURCE_DATA;
```

```
typedef struct D3D11_MAPPED_SUBRESOURCE {
    void *pData;
    UINT  RowPitch;
    UINT  DepthPitch;
} D3D11_MAPPED_SUBRESOURCE;
```

```
typedef enum D3D11_MAP {
    D3D11_MAP_READ          = 1,
    D3D11_MAP_WRITE         = 2,
    D3D11_MAP_READ_WRITE    = 3,
    D3D11_MAP_WRITE_DISCARD = 4,
    D3D11_MAP_WRITE_NO_OVERWRITE = 5
} D3D11_MAP;
```

```
typedef enum D3D11_MAP_FLAG {
    D3D11_MAP_FLAG_DO_NOT_WAIT = 0x100000L
} D3D11_MAP_FLAG;
```

```
typedef enum D3D11_RESOURCE_DIMENSION {
    D3D11_RESOURCE_DIMENSION_UNKNOWN = 0,
    D3D11_RESOURCE_DIMENSION_BUFFER = 1,
    D3D11_RESOURCE_DIMENSION_TEXTURE1D = 2,
    D3D11_RESOURCE_DIMENSION_TEXTURE2D = 3,
    D3D11_RESOURCE_DIMENSION_TEXTURE3D = 4
} D3D11_RESOURCE_DIMENSION;
```

ID3D11Texture1D

CD3D11_TEXTURE1D_DESC

```
typedef struct D3D11_TEXTURE1D_DESC {
    UINT          Width;
    UINT          MipLevels;
    UINT          ArraySize;
    DXGI_FORMAT   Format;
    D3D11_USAGE   Usage;
    UINT          BindFlags;
    UINT          CPUAccessFlags;
    UINT          MiscFlags;
} D3D11_TEXTURE1D_DESC;
```

ID3D11Texture2D

CD3D11_TEXTURE2D_DESC

```
typedef struct D3D11_TEXTURE2D_DESC {
    UINT          Width;
    UINT          Height;
    UINT          MipLevels;
    UINT          ArraySize;
    DXGI_FORMAT   Format;
    DXGI_SAMPLE_DESC SampleDesc;
    D3D11_USAGE   Usage;
    UINT          BindFlags;
    UINT          CPUAccessFlags;
    UINT          MiscFlags;
} D3D11_TEXTURE2D_DESC;
```

ID3D11Texture3D

CD3D11_TEXTURE3D_DESC

```
typedef struct D3D11_TEXTURE3D_DESC {
    UINT          Width;
    UINT          Height;
    UINT          Depth;
    UINT          MipLevels;
    DXGI_FORMAT   Format;
    D3D11_USAGE   Usage;
    UINT          BindFlags;
    UINT          CPUAccessFlags;
    UINT          MiscFlags;
} D3D11_TEXTURE3D_DESC;
```

ID3D11Buffer

CD3D11_BUFFER_DESC

```
typedef struct D3D11_BUFFER_DESC {
    UINT          ByteWidth;
    D3D11_USAGE   Usage;
    UINT          BindFlags;
    UINT          CPUAccessFlags;
    UINT          MiscFlags;
    UINT          StructureByteStride;
} D3D11_BUFFER_DESC;
```

ID3D11Asynchronous

ID3D11Counter

```
typedef struct D3D11_COUNTER_DESC {
    D3D11_COUNTER Counter;
    UINT          MiscFlags;
} D3D11_COUNTER_DESC;
```

```
typedef enum D3D11_COUNTER {
    D3D11_COUNTER_DEVICE_DEPENDENT_0 = 0x40000000
} D3D11_COUNTER;
```

ID3D11Predicate
ID3D11Query

```
typedef struct D3D11_QUERY_DESC {
    D3D11_QUERY Query;
    UINT          MiscFlags;
} D3D11_QUERY_DESC;
```

```
typedef enum D3D11_QUERY_MISC_FLAG {
    D3D11_QUERY_MISC_PREDICATEHINT = 0x1
} D3D11_QUERY_MISC_FLAG;
```

```
typedef struct D3D11_COUNTER_INFO {
    D3D11_COUNTER LastDeviceDependentCounter;
    UINT          NumSimultaneousCounters;
    UINT8         NumDetectableParallelUnits;
} D3D11_COUNTER_INFO;
```

```
typedef enum D3D11_QUERY {
    D3D11_QUERY_EVENT,
    D3D11_QUERY_OCCLUSION,
    D3D11_QUERY_TIMESTAMP,
    D3D11_QUERY_TIMESTAMP_DISJOINT,
    D3D11_QUERY_PIPELINE_STATISTICS,
    D3D11_QUERY_OCCLUSION_PREDICATE,
    D3D11_QUERY_SO_STATISTICS,
    D3D11_QUERY_SO_OVERFLOW_PREDICATE,
    D3D11_QUERY_SO_STATISTICS_STREAM0,
    D3D11_QUERY_SO_OVERFLOW_PREDICATE_STREAM0,
    D3D11_QUERY_SO_STATISTICS_STREAM1,
    D3D11_QUERY_SO_OVERFLOW_PREDICATE_STREAM1,
    D3D11_QUERY_SO_STATISTICS_STREAM2,
    D3D11_QUERY_SO_OVERFLOW_PREDICATE_STREAM2,
    D3D11_QUERY_SO_STATISTICS_STREAM3,
    D3D11_QUERY_SO_OVERFLOW_PREDICATE_STREAM3
} D3D11_QUERY;
```

Shaders

ID3D11ComputeShader
ID3D11DomainShader
ID3D11GeometryShader
ID3D11HullShader
ID3D11PixelShader
ID3D11VertexShader

```
typedef struct D3D11_SO_DECLARATION_ENTRY {  
    UINT    Stream;  
    LPCSTR  SemanticName;  
    UINT    SemanticIndex;  
    BYTE    StartComponent;  
    BYTE    ComponentCount;  
    BYTE    OutputSlot;  
} D3D11_SO_DECLARATION_ENTRY;
```

ID3D10Include
ID3DX11ThreadPump
ID3D11ClassLinkage
ID3D11ClassInstance
ID3D11ShaderReflection
ID3D11ShaderReflectionConstantBuffer
ID3D11ShaderReflectionType
ID3D11ShaderReflectionVariable

Views

ID3D11ShaderResourceView

CD3D11_SHADER_RESOURCE_VIEW_DESC

```
typedef struct D3D11_SHADER_RESOURCE_VIEW_DESC {
    DXGI_FORMAT          Format;
    D3D11_SRV_DIMENSION ViewDimension;
    union {
        D3D11_BUFFER_SRV      Buffer;
        D3D11_TEX1D_SRV       Texture1D;
        D3D11_TEX1D_ARRAY_SRV Texture1DArray;
        D3D11_TEX2D_SRV       Texture2D;
        D3D11_TEX2D_ARRAY_SRV Texture2DArray;
        D3D11_TEX2DMS_SRV      Texture2DMS;
        D3D11_TEX2DMS_ARRAY_SRV Texture2DMSArray;
        D3D11_TEX3D_SRV        Texture3D;
        D3D11_TEXCUBE_SRV      TextureCube;
        D3D11_TEXCUBE_ARRAY_SRV TextureCubeArray;
        D3D11_BUFFEREX_SRV     BufferEx;
    };
} D3D11_SHADER_RESOURCE_VIEW_DESC;
```

```
typedef enum D3D11_SRV_DIMENSION {
    D3D11_SRV_DIMENSION_UNKNOWN = 0,
    D3D11_SRV_DIMENSION_BUFFER  = 1,
    D3D11_SRV_DIMENSION_TEXTURE1D = 2,
    D3D11_SRV_DIMENSION_TEXTURE1DARRAY = 3,
    D3D11_SRV_DIMENSION_TEXTURE2D = 4,
    D3D11_SRV_DIMENSION_TEXTURE2DARRAY = 5,
    D3D11_SRV_DIMENSION_TEXTURE2DMS = 6,
    D3D11_SRV_DIMENSION_TEXTURE2DMSARRAY = 7,
    D3D11_SRV_DIMENSION_TEXTURE3D = 8,
    D3D11_SRV_DIMENSION_TEXTURECUBE = 9,
    D3D11_SRV_DIMENSION_TEXTURECUBEARRAY = 10,
    D3D11_SRV_DIMENSION_BUFFEREX = 11,
} D3D11_SRV_DIMENSION;
```

```
typedef enum D3D11_BUFFEREX_SRV_FLAG {
    D3D11_BUFFEREX_SRV_FLAG_RAW = 0x1
} D3D11_BUFFEREX_SRV_FLAG;
```

```
typedef struct D3D11_BUFFER_SRV {
    UINT ElementOffset;
    UINT ElementWidth;
} D3D11_BUFFER_SRV;
typedef struct D3D11_TEX1D_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
} D3D11_TEX1D_SRV;
typedef struct D3D11_TEX1D_ARRAY_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX1D_ARRAY_SRV;
typedef struct D3D11_TEX2D_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
} D3D11_TEX2D_SRV;
typedef struct D3D11_TEX2D_ARRAY_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2D_ARRAY_SRV;
typedef struct D3D11_TEX2DMS_SRV {
    UINT UnusedField_NothingToDefine;
} D3D11_TEX2DMS_SRV;
typedef struct D3D11_TEX2DMS_ARRAY_SRV {
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2DMS_ARRAY_SRV;
typedef struct D3D11_TEX3D_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
} D3D11_TEX3D_SRV;
typedef struct D3D11_TEXCUBE_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
} D3D11_TEXCUBE_SRV;
typedef struct D3D11_TEXCUBE_ARRAY_SRV {
    UINT MostDetailedMip;
    UINT MipLevels;
    UINT First2DArrayFace;
    UINT NumCubes;
} D3D11_TEXCUBE_ARRAY_SRV;
typedef struct D3D11_BUFFEREX_SRV {
    UINT FirstElement;
    UINT NumElements;
    UINT Flags;
} D3D11_BUFFEREX_SRV;
```

ID3D11RenderTargetView

CD3D11_RENDER_TARGET_VIEW_DESC

```
typedef struct D3D11_RENDER_TARGET_VIEW_DESC {
    DXGI_FORMAT          Format;
    D3D11_RTV_DIMENSION ViewDimension;
    union {
        D3D11_BUFFER_RTV      Buffer;
        D3D11_TEX1D_RTV       Texture1D;
        D3D11_TEX1D_ARRAY_RTV Texture1DArray;
        D3D11_TEX2D_RTV       Texture2D;
        D3D11_TEX2D_ARRAY_RTV Texture2DArray;
        D3D11_TEX2DMS_RTV     Texture2DMS;
        D3D11_TEX2DMS_ARRAY_RTV Texture2DMSArray;
        D3D11_TEX3D_RTV       Texture3D;
    };
} D3D11_RENDER_TARGET_VIEW_DESC;
```

```
typedef enum D3D11_RTV_DIMENSION {
    D3D11_RTV_DIMENSION_UNKNOWN      = 0,
    D3D11_RTV_DIMENSION_BUFFER      = 1,
    D3D11_RTV_DIMENSION_TEXTURE1D   = 2,
    D3D11_RTV_DIMENSION_TEXTURE1DARRAY = 3,
    D3D11_RTV_DIMENSION_TEXTURE2D   = 4,
    D3D11_RTV_DIMENSION_TEXTURE2DARRAY = 5,
    D3D11_RTV_DIMENSION_TEXTURE2DMS = 6,
    D3D11_RTV_DIMENSION_TEXTURE2DMSARRAY = 7,
    D3D11_RTV_DIMENSION_TEXTURE3D   = 8
} D3D11_RTV_DIMENSION;
```

```
typedef struct D3D11_BUFFER_RTV {
    UINT ElementOffset;
    UINT ElementWidth;
} D3D11_BUFFER_RTV;

typedef struct D3D11_TEX1D_RTV {
    UINT MipSlice;
} D3D11_TEX1D_RTV;

typedef struct D3D11_TEX1D_ARRAY_RTV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX1D_ARRAY_RTV;

typedef struct D3D11_TEX2D_RTV {
    UINT MipSlice;
} D3D11_TEX2D_RTV;

typedef struct D3D11_TEX2D_ARRAY_RTV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2D_ARRAY_RTV;

typedef struct D3D11_TEX2DMS_RTV {
    UINT UnusedField_NothingToDefine;
} D3D11_TEX2DMS_RTV;

typedef struct D3D11_TEX2DMS_ARRAY_RTV {
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2DMS_ARRAY_RTV;

typedef struct D3D11_TEX3D_RTV {
    UINT MipSlice;
    UINT FirstWSlice;
    UINT WSize;
} D3D11_TEX3D_RTV;
```


ID3D11DepthStencilView

CD3D11_DEPTH_STENCIL_VIEW_DESC

```
typedef struct D3D11_DEPTH_STENCIL_VIEW_DESC {
    DXGI_FORMAT          Format;
    D3D11_DSV_DIMENSION ViewDimension;
    UINT                 Flags;
    union {
        D3D11_TEX1D_DSV      Texture1D;
        D3D11_TEX1D_ARRAY_DSV Texture1DArray;
        D3D11_TEX2D_DSV      Texture2D;
        D3D11_TEX2D_ARRAY_DSV Texture2DArray;
        D3D11_TEX2DMS_DSV    Texture2DMS;
        D3D11_TEX2DMS_ARRAY_DSV Texture2DMSArray;
    };
} D3D11_DEPTH_STENCIL_VIEW_DESC;
```

```
typedef enum D3D11_DSV_DIMENSION {
    D3D11_DSV_DIMENSION_UNKNOWN = 0,
    D3D11_DSV_DIMENSION_TEXTURE1D = 1,
    D3D11_DSV_DIMENSION_TEXTURE1DARRAY = 2,
    D3D11_DSV_DIMENSION_TEXTURE2D = 3,
    D3D11_DSV_DIMENSION_TEXTURE2DARRAY = 4,
    D3D11_DSV_DIMENSION_TEXTURE2DMS = 5,
    D3D11_DSV_DIMENSION_TEXTURE2DMSARRAY = 6
} D3D11_DSV_DIMENSION;
```

```
typedef enum D3D11_DSV_FLAG {
    D3D11_DSV_READ_ONLY_DEPTH = 0x1L,
    D3D11_DSV_READ_ONLY_STENCIL = 0x2L
} D3D11_DSV_FLAG;
```

```
typedef struct D3D11_TEX1D_DSV {
    UINT MipSlice;
} D3D11_TEX1D_DSV;
```

```
typedef struct D3D11_TEX1D_ARRAY_DSV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX1D_ARRAY_DSV;
```

```
typedef struct D3D11_TEX2D_DSV {
    UINT MipSlice;
} D3D11_TEX2D_DSV;
```

```
typedef struct D3D11_TEX2D_ARRAY_DSV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2D_ARRAY_DSV;
```

```
typedef struct D3D11_TEX2DMS_DSV {
    UINT UnusedField_NothingToDefine;
} D3D11_TEX2DMS_DSV;
```

```
typedef struct D3D11_TEX2DMS_ARRAY_DSV {
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2DMS_ARRAY_DSV;
```

ID3D11UnorderedAccessView

CD3D11_UNORDERED_ACCESS_VIEW_DESC

```
typedef struct D3D11_UNORDERED_ACCESS_VIEW_DESC {
    DXGI_FORMAT          Format;
    D3D11_UAV_DIMENSION ViewDimension;
    union {
        D3D11_BUFFER_UAV      Buffer;
        D3D11_TEX1D_UAV       Texture1D;
        D3D11_TEX1D_ARRAY_UAV Texture1DArray;
        D3D11_TEX2D_UAV       Texture2D;
        D3D11_TEX2D_ARRAY_UAV Texture2DArray;
        D3D11_TEX3D_UAV       Texture3D;
    };
} D3D11_UNORDERED_ACCESS_VIEW_DESC;
```

```
typedef enum D3D11_UAV_DIMENSION {
    D3D11_UAV_DIMENSION_UNKNOWN      = 0,
    D3D11_UAV_DIMENSION_BUFFER       = 1,
    D3D11_UAV_DIMENSION_TEXTURE1D    = 2,
    D3D11_UAV_DIMENSION_TEXTURE1DARRAY = 3,
    D3D11_UAV_DIMENSION_TEXTURE2D    = 4,
    D3D11_UAV_DIMENSION_TEXTURE2DARRAY = 5,
    D3D11_UAV_DIMENSION_TEXTURE3D    = 6,
} D3D11_UAV_DIMENSION;
```

```
typedef enum D3D11_BUFFER_UAV_FLAG {
    D3D11_BUFFER_UAV_FLAG_RAW      = 0x1,
    D3D11_BUFFER_UAV_FLAG_APPEND  = 0x2,
    D3D11_BUFFER_UAV_FLAG_COUNTER = 0x4,
} D3D11_BUFFER_UAV_FLAG;
```

```
typedef struct D3D11_BUFFER_UAV {
    UINT FirstElement;
    UINT NumElements;
    UINT Flags;
} D3D11_BUFFER_UAV;

typedef struct D3D11_TEX1D_UAV {
    UINT MipSlice;
} D3D11_TEX1D_UAV;

typedef struct D3D11_TEX1D_ARRAY_UAV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX1D_ARRAY_UAV;

typedef struct D3D11_TEX2D_UAV {
    UINT MipSlice;
} D3D11_TEX2D_UAV;

typedef struct D3D11_TEX2D_ARRAY_UAV {
    UINT MipSlice;
    UINT FirstArraySlice;
    UINT ArraySize;
} D3D11_TEX2D_ARRAY_UAV;

typedef struct D3D11_TEX3D_UAV {
    UINT MipSlice;
    UINT FirstWSlice;
    UINT WSize;
} D3D11_TEX3D_UAV;
```

States

ID3D11BlendState

CD3D11_BLEND_DESC

```
typedef struct D3D11_BLEND_DESC {
    BOOL                AlphaToCoverageEnable;
    BOOL                IndependentBlendEnable;
    D3D11_RENDER_TARGET_BLEND_DESC RenderTarget[8];
} D3D11_BLEND_DESC;
```

```
typedef struct D3D11_RENDER_TARGET_BLEND_DESC {
    BOOL                BlendEnable;
    D3D11_BLEND        SrcBlend;
    D3D11_BLEND        DestBlend;
    D3D11_BLEND_OP     BlendOp;
    D3D11_BLEND        SrcBlendAlpha;
    D3D11_BLEND        DestBlendAlpha;
    D3D11_BLEND_OP     BlendOpAlpha;
    UINT8              RenderTargetWriteMask;
} D3D11_RENDER_TARGET_BLEND_DESC;
```

```
typedef enum D3D11_BLEND_OP {
    D3D11_BLEND_OP_ADD          = 1,
    D3D11_BLEND_OP_SUBTRACT    = 2,
    D3D11_BLEND_OP_REV_SUBTRACT = 3,
    D3D11_BLEND_OP_MIN         = 4,
    D3D11_BLEND_OP_MAX         = 5
} D3D11_BLEND_OP;
```

```
typedef enum D3D11_BLEND {
    D3D11_BLEND_ZERO          = 1,
    D3D11_BLEND_ONE           = 2,
    D3D11_BLEND_SRC_COLOR     = 3,
    D3D11_BLEND_INV_SRC_COLOR = 4,
    D3D11_BLEND_SRC_ALPHA     = 5,
    D3D11_BLEND_INV_SRC_ALPHA = 6,
    D3D11_BLEND_DEST_ALPHA    = 7,
    D3D11_BLEND_INV_DEST_ALPHA = 8,
    D3D11_BLEND_DEST_COLOR    = 9,
    D3D11_BLEND_INV_DEST_COLOR = 10,
    D3D11_BLEND_SRC_ALPHA_SAT = 11,
    D3D11_BLEND_BLEND_FACTOR  = 14,
    D3D11_BLEND_INV_BLEND_FACTOR = 15,
    D3D11_BLEND_SRC1_COLOR    = 16,
    D3D11_BLEND_INV_SRC1_COLOR = 17,
    D3D11_BLEND_SRC1_ALPHA    = 18,
    D3D11_BLEND_INV_SRC1_ALPHA = 19
} D3D11_BLEND;
```

ID3D11DepthStencilState

CD3D11_DEPTH_STENCIL_DESC

```
typedef struct D3D11_DEPTH_STENCIL_DESC {
    BOOL                DepthEnable;
    D3D11_DEPTH_WRITE_MASK    DepthWriteMask;
    D3D11_COMPARISON_FUNC    DepthFunc;
    BOOL                StencilEnable;
    UINT8               StencilReadMask;
    UINT8               StencilWriteMask;
    D3D11_DEPTH_STENCIL_OP_DESC    FrontFace;
    D3D11_DEPTH_STENCIL_OP_DESC    BackFace;
} D3D11_DEPTH_STENCIL_DESC;
```

```
typedef struct D3D11_DEPTH_STENCIL_OP_DESC {
    D3D11_STENCIL_OP    StencilFailOp;
    D3D11_STENCIL_OP    StencilDepthFailOp;
    D3D11_STENCIL_OP    StencilPassOp;
    D3D11_COMPARISON_FUNC    StencilFunc;
} D3D11_DEPTH_STENCIL_OP_DESC;
```

```
typedef enum D3D11_DEPTH_WRITE_MASK {
    D3D11_DEPTH_WRITE_MASK_ZERO    = 0,
    D3D11_DEPTH_WRITE_MASK_ALL     = 1
} D3D11_DEPTH_WRITE_MASK;
```

```
typedef enum D3D11_COMPARISON_FUNC {
    D3D11_COMPARISON_NEVER         = 1,
    D3D11_COMPARISON_LESS         = 2,
    D3D11_COMPARISON_EQUAL        = 3,
    D3D11_COMPARISON_LESS_EQUAL   = 4,
    D3D11_COMPARISON_GREATER      = 5,
    D3D11_COMPARISON_NOT_EQUAL    = 6,
    D3D11_COMPARISON_GREATER_EQUAL = 7,
    D3D11_COMPARISON_ALWAYS       = 8
} D3D11_COMPARISON_FUNC;
```

ID3D11RasterizerState

CD3D11_RASTERIZER_DESC

```
typedef struct D3D11_RASTERIZER_DESC {
    D3D11_FILL_MODE    FillMode;
    D3D11_CULL_MODE    CullMode;
    BOOL               FrontCounterClockwise;
    INT                 DepthBias;
    FLOAT               DepthBiasClamp;
    FLOAT               SlopeScaledDepthBias;
    BOOL               DepthClipEnable;
    BOOL               ScissorEnable;
    BOOL               MultisampleEnable;
    BOOL               AntialiasedLineEnable;
} D3D11_RASTERIZER_DESC;
```

```
typedef enum D3D11_STENCIL_OP {
    D3D11_STENCIL_OP_KEEP         = 1,
    D3D11_STENCIL_OP_ZERO        = 2,
    D3D11_STENCIL_OP_REPLACE     = 3,
    D3D11_STENCIL_OP_INCR_SAT    = 4,
    D3D11_STENCIL_OP_DECR_SAT   = 5,
    D3D11_STENCIL_OP_INVERT     = 6,
    D3D11_STENCIL_OP_INCR       = 7,
    D3D11_STENCIL_OP_DECR       = 8
} D3D11_STENCIL_OP;
```

```
typedef enum D3D11_CULL_MODE {
    D3D11_CULL_NONE    = 1,
    D3D11_CULL_FRONT   = 2,
    D3D11_CULL_BACK    = 3
} D3D11_CULL_MODE;
```

```
typedef enum D3D11_FILL_MODE {
    D3D11_FILL_WIREFRAME    = 2,
    D3D11_FILL_SOLID        = 3
} D3D11_FILL_MODE;
```

ID3D11SamplerState

CD3D11_SAMPLER_DESC

```
typedef struct D3D11_SAMPLER_DESC {
    D3D11_FILTER          Filter;
    D3D11_TEXTURE_ADDRESS_MODE AddressU;
    D3D11_TEXTURE_ADDRESS_MODE AddressV;
    D3D11_TEXTURE_ADDRESS_MODE AddressW;
    FLOAT                 MipLODBias;
    UINT                  MaxAnisotropy;
    D3D11_COMPARISON_FUNC ComparisonFunc;
    FLOAT                 BorderColor[4];
    FLOAT                 MinLOD;
    FLOAT                 MaxLOD;
} D3D11_SAMPLER_DESC;
```

```
typedef enum D3D11_TEXTURE_ADDRESS_MODE {
    D3D11_TEXTURE_ADDRESS_WRAP          = 1,
    D3D11_TEXTURE_ADDRESS_MIRROR       = 2,
    D3D11_TEXTURE_ADDRESS_CLAMP        = 3,
    D3D11_TEXTURE_ADDRESS_BORDER       = 4,
    D3D11_TEXTURE_ADDRESS_MIRROR_ONCE  = 5
} D3D11_TEXTURE_ADDRESS_MODE;
```

```
typedef enum D3D11_FILTER_TYPE {
    D3D11_FILTER_TYPE_POINT   = 0,
    D3D11_FILTER_TYPE_LINEAR  = 1
} D3D11_FILTER_TYPE;
```

```
typedef enum D3D11_FILTER {
    D3D11_FILTER_MIN_MAG_MIP_POINT
    D3D11_FILTER_MIN_MAG_POINT_MIP_LINEAR
    D3D11_FILTER_MIN_POINT_MAG_LINEAR_MIP_POINT
    D3D11_FILTER_MIN_POINT_MAG_MIP_LINEAR
    D3D11_FILTER_MIN_LINEAR_MAG_MIP_POINT
    D3D11_FILTER_MIN_LINEAR_MAG_POINT_MIP_LINEAR
    D3D11_FILTER_MIN_MAG_LINEAR_MIP_POINT
    D3D11_FILTER_MIN_MAG_MIP_LINEAR
    D3D11_FILTER_ANISOTROPIC
    D3D11_FILTER_COMPARISON_MIN_MAG_MIP_POINT
    D3D11_FILTER_COMPARISON_MIN_MAG_POINT_MIP_LINEAR
    D3D11_FILTER_COMPARISON_MIN_POINT_MAG_LINEAR_MIP_POINT
    D3D11_FILTER_COMPARISON_MIN_POINT_MAG_MIP_LINEAR
    D3D11_FILTER_COMPARISON_MIN_LINEAR_MAG_MIP_POINT
    D3D11_FILTER_COMPARISON_MIN_LINEAR_MAG_POINT_MIP_LINEAR
    D3D11_FILTER_COMPARISON_MIN_MAG_LINEAR_MIP_POINT
    D3D11_FILTER_COMPARISON_MIN_MAG_MIP_LINEAR
    D3D11_FILTER_COMPARISON_ANISOTROPIC
    D3D11_FILTER_TEXT_1BIT
} D3D11_FILTER;
```

HLSL

register

: register ([shader_profile], Type#[subcomponent])

Semantics

D3D9 and 10, Vertex Shader input:

BINORMAL[n]	float4
BLENDINDICES[n]	uint
BLENDWEIGHT[n]	float
COLOR[n]	float4
NORMAL[n]	float4
POSITION[n]	float4
POSITIONT	float4
PSIZE[n]	float
TANGENT[n]	float4
TEXCOORD[n]	float4

D3D9 and 10, Vertex Shader output:

COLOR[n]	float4
FOG	float
POSITION[n]	float4
PSIZE	float
TESSFACTOR[n]	float
TEXCOORD[n]	float4

D3D9 and 10, Pixel Shader input:

COLOR[n]	float4
TEXCOORD[n]	float4
VFACE	float
VPOS	float2

D3D9 and 10, Pixel Shader output:

COLOR[n]	float4
DEPTH[n]	float

Type:

- b Constant buffer
- t Texture and texture buffer
- c Buffer offset
- s Sampler
- u Unordered Access View

System-Value Semantics (D3D10):

SV_ClipDistance[n]	float
SV_CullDistance[n]	float
SV_Coverage	bool
SV_Depth	float
SV_DispatchThreadID	uint3
SV_DomainLocation	float2 3
SV_GroupID	uint3
SV_GroupIndex	uint
SV_GroupThreadID	uint3
SV_GSInstanceID	uint
SV_InsideTessFactor	float float[2]
SV_IsFrontFace	bool
SV_OutputControlPointID	uint
SV_Position	float4
SV_RenderTargetArrayIndex	uint
SV_SampleIndex	uint
SV_Target[n]	float
SV_TessFactor	float[2 3 4]
SV_ViewportArrayIndex	uint
SV_InstanceID	uint
SV_PrimitiveID	uint
SV_VertexID	uint

FXC Parameters

Usage: fxc <options> <files>

/?, /help	print this message	/Ni	output instruction numbers in assembly listings
/T<profile>	target profile	/P<file>	preprocess to file (must be used alone)
/E<name>	entrypoint name	@<file>	options response file
/I<include>	additional include path	/dumpbin	load a binary file rather than compiling
/Vi	display details about the include process	/Qstrip_reflect	strip reflection data from 4_0+ shader bytecode
/Od	disable optimizations	/Qstrip_debug	strip debug information from 4_0+ shader bytecode
/Op	disable preshadings	/compress	compress DX10 shader bytecode from files
/O{0,1,2,3}	optimization level 0..3. 1 is default	/decompress	decompress bytecode from first file, output files should be listed in the order they were in during compression
/WX	treat warnings as errors	/D<id>=<text>	define macro
/Vd	disable validation	/LD	Load d3dx9_31.dll
/Zi	enable debugging information	/nologo	suppress copyright message
/Zpr	pack matrices in row-major order		
/Zpc	pack matrices in column-major order		
/Gpp	force partial precision		
/Gfa	avoid flow control constructs		
/Gfp	prefer flow control constructs		
/Gdp	disable effect performance mode		
/Ges	enable strict mode		
/Gec	enable backwards compatibility mode		
/Gis	force IEEE strictness		
/Gch	compile as a child effect for FX 4.x targets		
/Fo<file>	output object file		
/Fc<file>	output assembly code listing file		
/Fh<file>	output assembly code and hex listing file		
/Fh<file>	output header file containing object code		
/Fe<file>	output warnings and errors to a specific file		
/Vn<name>	use <name> as variable name in header file		
/Cc	output color coded assembly listings		

<profile>: cs_4_0 cs_4_1 cs_5_0 ds_5_0 fx_2_0 fx_4_0 fx_4_1 fx_5_0 gs_4_0 gs_4_1
gs_5_0 hs_5_0 ps_2_0 ps_2_a ps_2_b ps_2_sw ps_3_0 ps_3_sw ps_4_0 ps_4_0_level_9_1
ps_4_0_level_9_3 ps_4_0_level_9_0 ps_4_1 ps_5_0 tx_1_0 vs_1_1 vs_2_0 vs_2_a vs_2_sw
vs_3_0 vs_3_sw vs_4_0 vs_4_0_level_9_1 vs_4_0_level_9_3 vs_4_0_level_9_0 vs_4_1
vs_5_0