다이렉트 X

#2. 다이렉트 X를 이용한 2차원 그래픽 프로그래밍

이강훈
서울대학교 전기컴퓨터공학부
3MAP 연구실

Part I : 버텍스와 프리미티브

- 그림 그릴 도구
  - 버텍스(Vertex)
  - 프리미티브(Primitive)
  - 2차원 그림 그리기

그림 그릴 도구

- 포토샵의 예

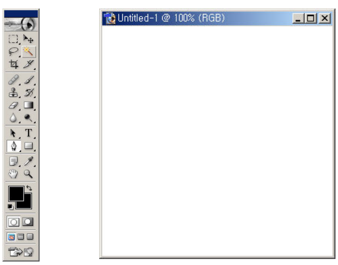


그림 그릴 도구

- 포토샵의 예

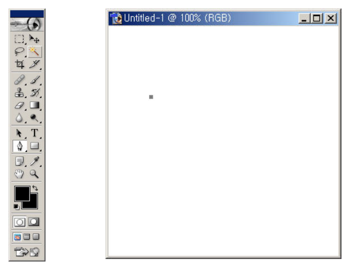


그림 그릴 도구

- 포토샵의 예

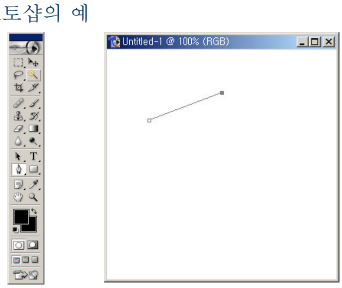


그림 그릴 도구

- 포토샵의 예

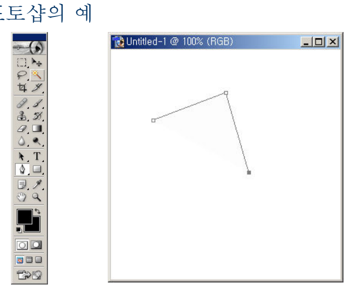


그림 그릴 도구

- 포토샵의 예

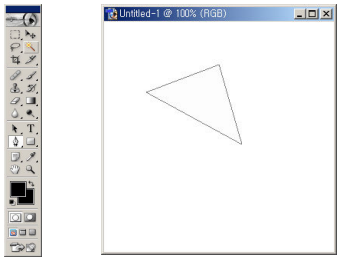


그림 그릴 도구

- 그렇다면 다이렉트X는?
  - 버텍스(vertex)
    - 점
  - 프리미티브(primitive)
    - 점 : 한 개의 점
    - 선 : 두 개의 점
    - 삼각형 : 세 개의 점

그림 그릴 도구

- 다이렉트X의 2차원 좌표계

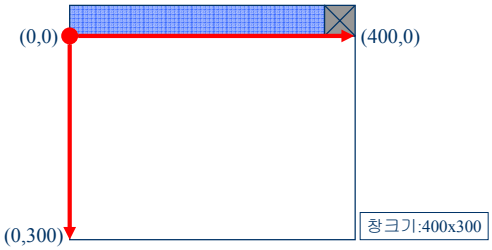


그림 그릴 도구

- 버텍스 배치

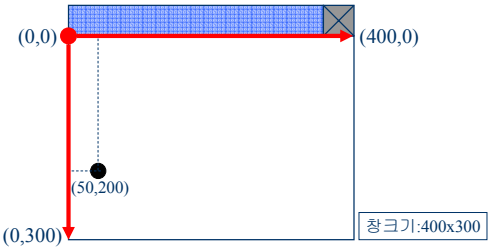


그림 그릴 도구

- 버텍스 배치

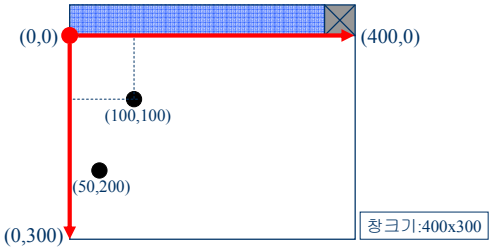


그림 그릴 도구

- 버텍스 배치

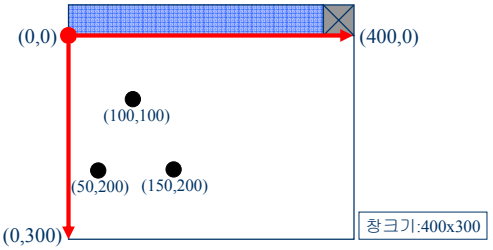


그림 그릴 도구

- 버텍스 배치

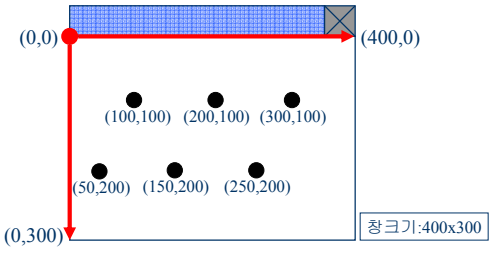


그림 그릴 도구

- 버텍스와 프리미티브
  - 버텍스
    - 실제 그림이 아니다
    - 프리미티브를 구성하는 일부 즉, 도구
  - 프리미티브
    - 실제 그림이다
    - 여러 개의 버텍스를 묶어서 하나의 그림으로 만듦
    - 버텍스를 어떻게 해석하느냐에 따라 점, 선, 삼각형 모양을 이룸

그림 그릴 도구

- 프리미티브: 점(Point)으로 해석

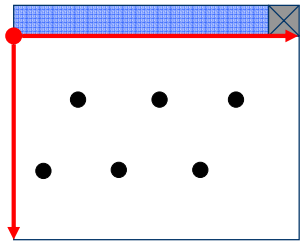


그림 그릴 도구

- 프리미티브: 선(Line)으로 해석

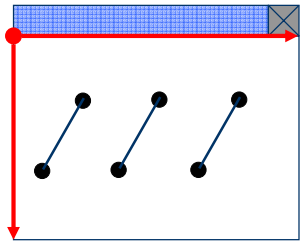


그림 그릴 도구

- 프리미티브: 연결된 선(Line Strip)으로 해석

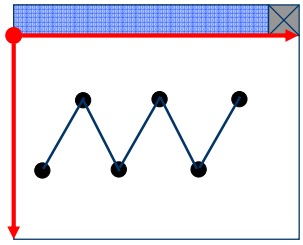


그림 그릴 도구

- 프리미티브: 삼각형(Triangle)으로 해석

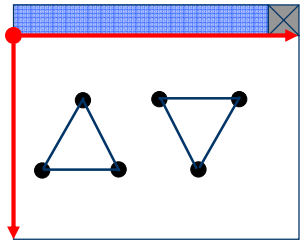


그림 그릴 도구

- 프리미티브: 연결된 삼각형(Triangle Strip)으로 해석

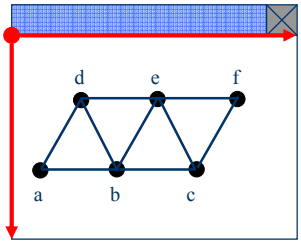


그림 그릴 도구

- 그렇다면 코딩은?
  1. 버텍스 정보를 담을 구조체 정의
  2. 버텍스 집합을 배열에 저장
  3. 버텍스 집합을 버텍스 버퍼에 복사
  4. 프리미티브 그리기

그림 그릴 도구

- 그렇다면 코딩은?

1. 구조체 정의

```
struct Vertex {
FLOAT x;
FLOAT y;
FLOAT z;
FLOAT rhw;
};
```

그림 그릴 도구

- 그렇다면 코딩은?

1. 구조체 정의

```
struct Vertex {
FLOAT x;
FLOAT y;
FLOAT z;
FLOAT rhw;
};
```

2. 배열에 저장

```
Vertex[0] = ;
Vertex[1] = ;
Vertex[2] = ;
Vertex[3] = ;
Vertex[4] = ;
Vertex[5] = ;
```

그림 그릴 도구

- 그렇다면 코딩은?

1. 구조체 정의

```
struct Vertex {
FLOAT x;
FLOAT y;
FLOAT z;
FLOAT rhw;
};
```

2. 배열에 저장

```
Vertex[0] = ;
Vertex[1] = ;
Vertex[2] = ;
Vertex[3] = ;
Vertex[4] = ;
Vertex[5] = ;
```

3. 버퍼에 복사

→ VertexBuffer

그림 그릴 도구

- 그렇다면 코딩은?

1. 구조체 정의

```
struct Vertex {
FLOAT x;
FLOAT y;
FLOAT z;
FLOAT rhw;
};
```

2. 배열에 저장

```
Vertex[0] = ;
Vertex[1] = ;
Vertex[2] = ;
Vertex[3] = ;
Vertex[4] = ;
Vertex[5] = ;
```

3. 버퍼에 복사

VertexBuffer →

4. 그리기

DrawPrimitive

그림 그릴 도구

- 버텍스 버퍼?
  - 배열과 버텍스 버퍼에 2중으로 저장하는 이유가 무엇인가?
  - 버텍스 버퍼에 저장하면 더 빠른 속도를 얻을 수 있다!

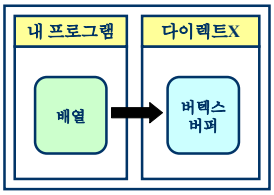


그림 그릴 도구

- 1단계: 구조체를 정의하자

```
struct Vertex
{
    FLOAT x;            // !
    FLOAT y;            // !
    FLOAT z;
    FLOAT rhw;
    DWORD color;        // !
};
#define D3DFVF_VERTEX ( D3DFVF_XYZRHW | D3DFVF_DIFFUSE )
```

그림 그릴 도구

- 2단계: 배열에 버텍스를 저장하자!

```
Vertex vertices[6];

vertices[0].x = 50.0F;      vertices[0].y = 200.0F;
vertices[0].z = 0.5F;       vertices[0].rhw = 1.0F;
vertices[0].color = D3DCOLOR_RGBA(0, 255, 0);

vertices[1].x = 100.0F;     vertices[1].y = 100.0F;
vertices[1].z = 0.5F;       vertices[1].rhw = 1.0F;
vertices[1].color = D3DCOLOR_RGBA(0, 255, 0);
// ...
```

그림 그릴 도구

- 3단계: 버텍스 버퍼를 만들자!

```
IDirect3DVertexBuffer9* vertex_buffer = 0;

d3d9_device->CreateVertexBuffer(
    sizeof(Vertex) * 6,             // ! ( == sizeof(vertices) )
    0,
    D3DFVF_VERTEX,                  // !
    D3DPOOL_DEFAULT,
    &vertex_buffer,                 // !
    0
);
```

그림 그릴 도구

- 4단계: 배열을 버텍스 버퍼로 복사하자!

```
VOID* vbuf_ptr = 0;
vertex_buffer->Lock(
    0,
    sizeof(Vertex) * 6,
    &vbuf_ptr,
    0
);
CopyMemory( vbuf_ptr, vertices, sizeof(Vertex) * 6 );
vertex_buffer->Unlock();
```

그림 그릴 도구

- 5단계: 버텍스 버퍼의 내용을 그리자!

```
d3d9_device->SetStreamSource(0, vertex_buffer, 0, sizeof(Vertex) );
d3d9_device->SetFVF( D3DFVF_VERTEX );
d3d9_device->DrawPrimitive(
    D3DPT_TRIANGLELIST,             // ! (*)
    0,                              // !
    2                               // !
);

(*) D3DPT_POINTLIST, D3DPT_LINELIST, D3DPT_LINESTRIP,
    D3DPT_TRIANGLELIST, D3DPT_TRIANGLESTRIP,
    D3DPT_TRIANGLEFAN
```

그림 그릴 도구

- 6단계: 잊지 말아야 할 것! 버텍스 버퍼 소멸!!!

```
void finalize()
{
    if( vertex_buffer )
    {
        vertex_buffer->Release();
    }
    // ...
}
```

직접 해 보기!!!

- 해 볼 것
  - 버텍스 여섯 개를 사용해서 그림 그리기
    - 제공한 source 파일의 내용 채우기
    - 여러 가지 프리미티브로 실험해보기
    - 각 버텍스의 색깔 다르게 지정해보기
    - 버텍스의 위치 바꿔보기

Part II : 2차원 벡터 그래픽스

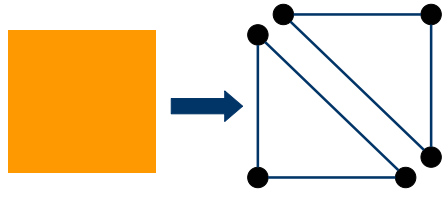
- 여러 가지 모양 만들기
  - 사각형
  - 원

여러 가지 모양 만들기

- 사각형
  - D3DPT_TRIANGLELIST 를 이용하면?
    - 버텍스 개수 : 6개
    - 프리미티브 개수 : 2개
  - D3DPT_TRIANGLESTRIP 을 이용하면?
    - 버텍스 개수 : 4개
    - 프리미티브 개수 : 2개

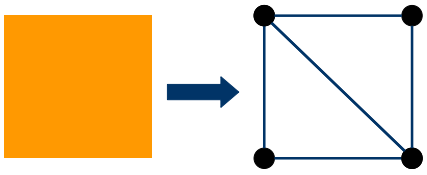
여러 가지 모양 만들기

- 사각형
  - D3DPT_TRIANGLELIST



여러 가지 모양 만들기

- 사각형
  - D3DPT_TRIANGLESTRIP



여러 가지 모양 만들기

- 원

여러 가지 모양 만들기

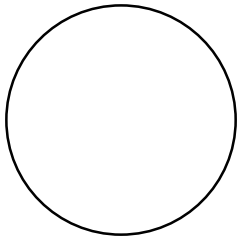
- 원
  - 다이렉트X 는 점, 선, 삼각형 모양밖에 지원하지 않는다
  - 어떻게 삼각형으로 원을 만들까?

정답

삼각형으로 쪼갠다!

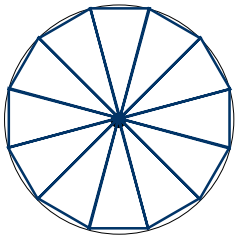
여러 가지 모양 만들기

- 원



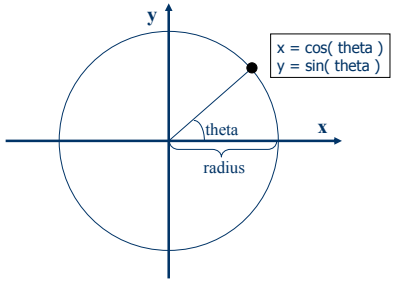
여러 가지 모양 만들기

- 원



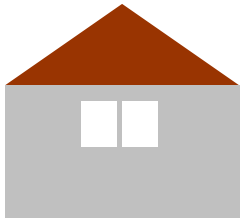
여러 가지 모양 만들기

- 원



모양 조합하기

- 삼각형과 여러 개의 사각형 조합해서 집 만들기

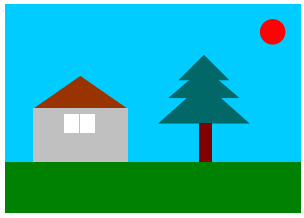


모양 조합하기

- 여러 개의 삼각형과 사각형 조합해서 나무 만들기

직접 해 볼 것

- 해 볼 것
  - 하나의 완성된 장면 구성하기
    - 풀밭
    - 나무
    - 집
    - 태양
    - 하늘



Part III : 비트맵 이미지와 텍스트

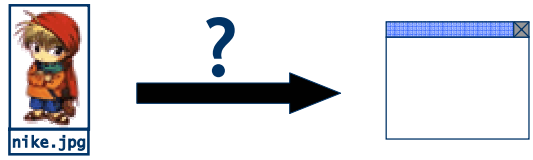
- 이미지 넣기
  - 스프라이트와 텍스처
  - Draw 함수의 인자 조절
- 글자 넣기

이미지 넣기

- 이미지?
  - 2D 비트맵 이미지
    - bmp, tga, jpg 등 여러 가지 포맷
- 이미지를 어떻게 넣나?
  - 3단계
    - 스프라이트 객체를 생성한다
    - 이미지 파일을 읽어들여 텍스처 객체를 생성한다
    - 스프라이트 객체를 이용해서 텍스처 객체를 화면에 그린다

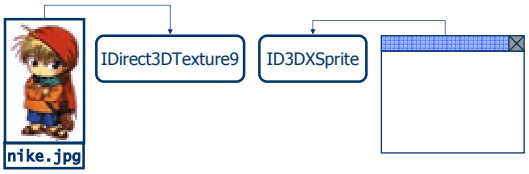
이미지 넣기

- 스프라이트와 텍스처를 이용한 이미지 삽입



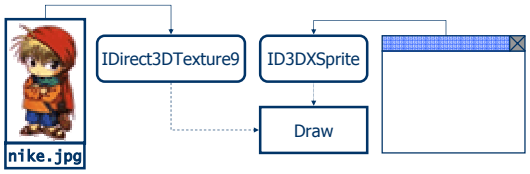
이미지 넣기

- 스프라이트와 텍스처를 이용한 이미지 삽입



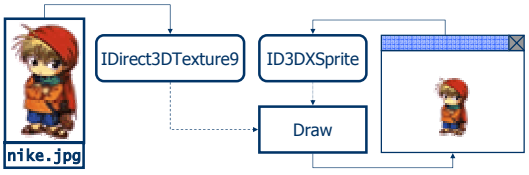
이미지 넣기

- 스프라이트와 텍스처를 이용한 이미지 삽입



이미지 넣기

- 스프라이트와 텍스처를 이용한 이미지 삽입



이미지 넣기

- 코딩은 어떻게?
  - 1단계: 디바이스로부터 스프라이트 객체 생성하기

```
ID3DXSprite* sprite = 0;
D3DXCreateSprite( d3d9_device, &sprite );
```

이미지 넣기

- 코딩은 어떻게?
  - 2단계: 이미지 파일로부터 텍스처 객체 생성하기

```
IDirect3DTexture9* texture = 0;
D3DXCreateTextureFromFileEx(
    d3d9_device,
    "nike.jpg",
    // ...
    &texture
);
```

이미지 넣기

- 코딩은 어떻게?
  - 3단계: 스프라이트 객체의 Draw 함수에 텍스처 객체 넘겨주기

```
HRESULT ID3DXSprite::Draw(
    LPDIRECT3DTEXTURE9 pSrcTexture,
    CONST RECT *pSrcRect,
    CONST D3DXVECTOR2 *pScaling,
    CONST D3DXVECTOR2 *pRotationCenter, FLOAT Rotation,
    CONST D3DVECTOR2 *pTranslation,
    D3DCOLOR Color
);
```

이미지 넣기

- 코딩은 어떻게?
  - 3단계: 스프라이트 객체의 Draw 함수에 텍스처 객체 넘겨주기

```
RECT image_rect={0, 0, 100, 160};
sprite->Draw(
    texture,
    &image_rect,
    NULL,
    NULL, 0.0F,
    NULL,
    D3DCOLOR_RGBA(255, 255, 255, 255)
);
```

이미지 넣기

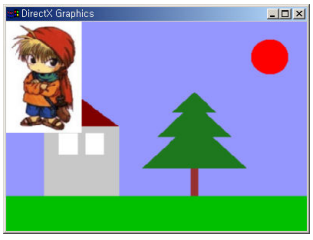
- 코딩은 어떻게?
  - 4단계: 생성된 스프라이트, 텍스처 객체 소멸 !!!

```
if( sprite ) {
    sprite->Release();
}

if( texture ) {
    texture->Release();
}
```

이미지 넣기

- 결과



이미지 넣기

- 좀더 보기좋게 하려면...
  - 니케의 크기를 더 작게 하고 싶다
  - 니케를 나무 옆, 풀밭 위에 서있게 하고 싶다

해결책

Draw의 인자를 이용한다!

이미지 넣기

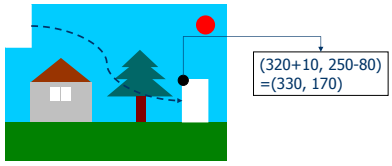
- 니케의 크기를 더 작게 하고 싶다
  - 니케의 원래 크기는?
    - 가로: 100, 세로: 160
  - 우리가 원하는 크기는?
    - 원래 크기의 50%(가로, 세로 모두)
    - 가로: 50, 세로: 80
  - Draw의 세 번째 인자를 이용한다

```
D3DXVECTOR2 scaling(0.5F, 0.5F);
```

이미지 넣기

- 니케의 위치를 옮기고 싶다
  - Draw의 여섯 번째 인자를 이용한다

```
D3DXVECTOR2 translation(330.0F, 170.0F);
```



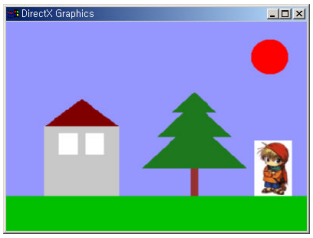
이미지 넣기

- 수정된 Draw 코드

```
RECT image_rect={0, 0, 100, 160};
D3DXVECTOR2 scaling( 0.5F, 0.5F );
D3DXVECTOR2 translation( 330.0F, 170.0F );
sprite->Draw(
    texture, &image_rect,
    &scaling,
    NULL, 0.0F,
    &translation,
    D3DCOLOR_RGBA(255, 255, 255, 255)
);
```

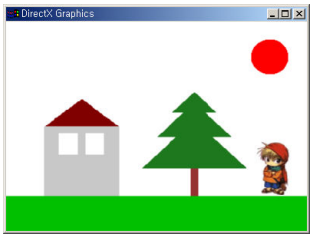
이미지 넣기

- 수정된 결과



이미지 넣기

- 수정된 결과



글자 넣기

- 1단계
  - ID3DXFont 객체를 생성한다

```
ID3DXFont* font = 0;
D3DXCreateFont(
      d3d9_device,
      (HFONT)::GetStockObject(ANSI_VAR_FONT),
      &font
);
```

글자 넣기

- 2단계
  - ID3DXFont::DrawText 함수를 호출한다

```
RECT text_rect = {0, 0, 400, 300};
font->DrawText(
    "안녕하세요",
    10,
    &text_rect,
    0,
    D3DCOLOR_RGBA(0, 0, 0, 255)
);
```

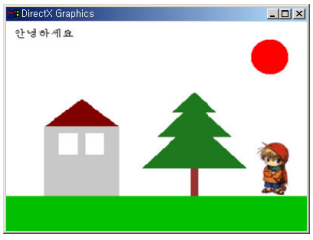
글자 넣기

- 3단계
  - 생성된 폰트 객체 소멸 !!!

```
if( font ) {
    font->Release();
}
```

글자 넣기

- 결과



직접 해 볼 것

- 이미지 넣기
  - 이미지 회전시켜보기
    - WM_TIMER 메시지 이용
    - Draw 함수의 네 번째 인자와 다섯 번째 인자 조절
      - 네 번째 인자 : 회전의 중심 좌표 (X:25.0F, Y: 40.0F)
      - 다섯 번째 인자 : 회전의 각도 (시간에 따라 증가시키기)
- 글자 넣기
  - 한 화면에 여러 개의 텍스트 배치해보기
    - RECT text_rect = { /*...*/ }