2D Game Programming 03

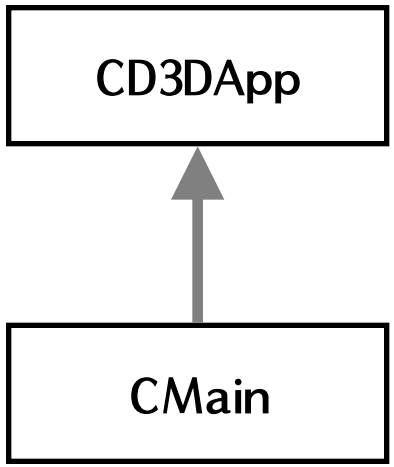
afewhee@gmail.com

1. C++ class 디자인

- CD3DApp
  - 윈도우, D3D, Device, Sprite 생성, 소멸
  - 시간, 후면 버퍼, Default 메시지 처리등 시스템에서 필요로 하는 기능 수행

- CMain
  - CD3DApp 를 상속 받은 클래스
  - Game Application에 관련된 인풋, 사운드, 네트워크, 게임 위상을 관리하는 기능 수행

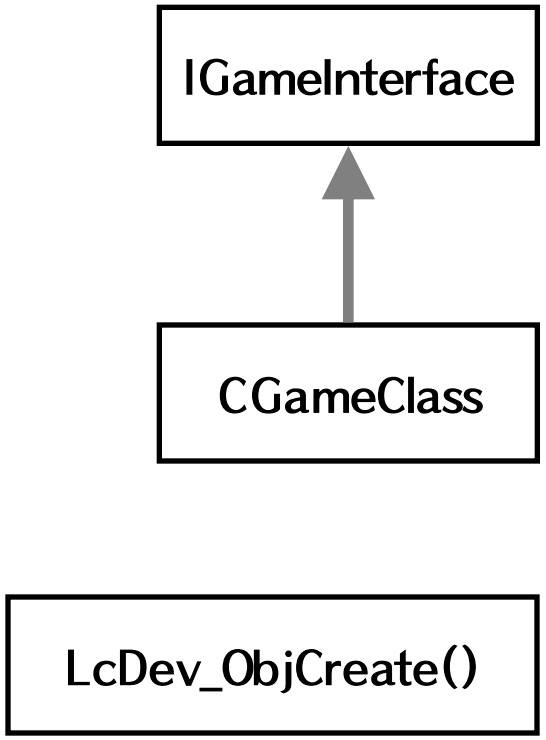
- 렌더링 클래스 공통 함수
  - 함수의 이름을 일정하게 두어 일관성을 유지하도록 함
  - 소멸자: 반드시 virtual로 선언해서 사용
  - 초기화: virtual INT Init()
  - 생 성: virtual INT Create()
  - 소 멸: virtual void Destroy()
  - 렌더링: virtual void Render()
  - 갱 신: virtual INT FrameMove()



1. C++ class 디자인

- 추상화가 필요한 주요 기본 클래스
  - 텍스처 ILcTex
  - 폰트 ILcFont
  - 인풋 ILcInput
  - 사운드, 미디 ILcSmd
  - 네트워크 ILcNet
  - 모델 ILcMdl

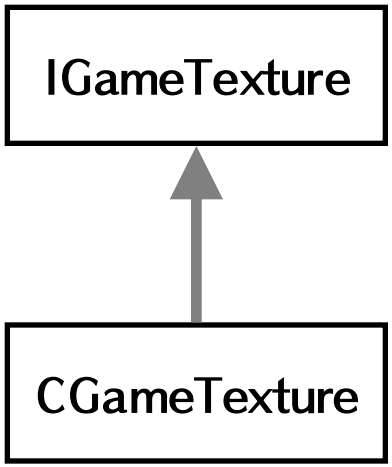
- 추상 클래스를 상속 받은 객체는 함수를 통해서 Instance를 생성
  - LcDev_TextureCreate()
  - LcDev_FontCreate()
  - LcDev_SmdCreate()
  - LcDev_MdlCreate()



2. Game Texture class

- 텍스처 class:
  - Direct3DTexture는 텍스처의 크기를 2의 승수로 정함
  - 이미지의 크기 등에 정보를 저장할 필요가 있음

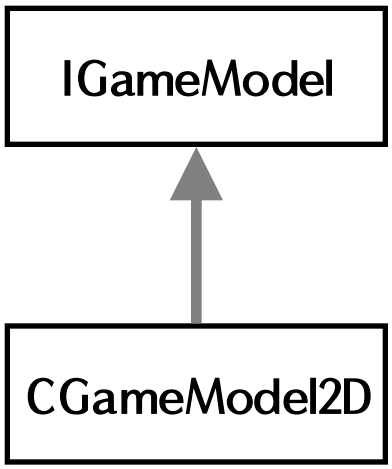
```
Ex)
interface IGameTexture
{
        virtual ~IGameTexture(){}
        virtual  int     Create(void* pDev, void* )=0;
        virtual  void  Destroy()=0;
        virtual  int     GetImageWidth() =0;
        virtual  int     GetImageHeight() =0;
        virtual  void* GetTexture() =0;
};

int LpDev_CreateTexture("{file|resource}"
                        , void* pDev
                        , void* {TextureName|ReourceID});

class CGameTexture : public IGameTexture
{
protected:
      LPDIRECT3DTEXTURE9 m_pTex;
      D3DXIMAGE_INFO       m_pInf;

  public:
      CGameTexture();
      virtual ~ CGameTexture();

      int     Create(void* pDev, void* TextureName);
      void   Destroy();

      int     GetImageWidth() { return m_pInf.Width; }
      int     GetImageHeight(){ return m_pInf.Height; }
      void*  GetTexture()      { return m_pTex;        }
};
```



- Texture Animation Class
  - 시간, 텍스처, 이미지 영역
  - 파이에서 로딩

Ex)

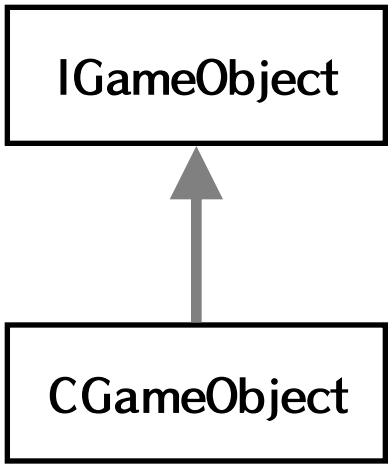
```
interface IGameModel
{
	virtual ~IGameModel(){};

	virtual int		Create(void* pSprite, void* ModelName) = 0;
	virtual void		Destroy() = 0;
	virtual int		FrameMove() = 0;
	virtual void		Render() = 0;
	virtual void		Play() = 0;
	virtual void		Stop() = 0;

	virtual  void		SetPos(const FLOAT*)=0;
	virtual  void		SetColor(const DWORD)=0;
};

class CGameModel2D : public IGameModel
{
protected:
	DWORD m_dTimeDelta;		// 시간 간격
	DWORD m_dTimeBgn;		// 시작 타임
	DWORD m_dTimeCur;		// 현재 타임
	INT	m_nAniTot;		// Ani Total Number
	INT	m_nAniCur;		// Current Ani Index

public:
	CGameModel2D();
	virtual ~CGameModel2D();
	...
```



- GameObject class
  - Game Data: 온라인게임을 만들었을 때 서버와 주고 받는정보
  - GameModel: 렌더링 데이터. 서버와 무관한 정보

Ex)

```
interface IGameObject
{
    virtual ~IGameObject(){};

    virtual int     Create() = 0;
    virtual void    Destroy() = 0;

    virtual int     FrameMove() = 0;
    virtual void    Render() = 0;

    virtual         int         SetModel(char* sModel)=0;
    virtual         void        SetPos(const FLOAT*)=0;
};
```




```
class CGameObject : public IGameObject
{
protected:
    D3DXVECTOR3             m_vcPos;
    IGameModel*             m_pModel;

public:
    ...
};
```

5. Game Input class

- Keyboard
  - GetAsyncKeystate()
    - Virtual Key의 값을 즉시 알아냄
    - ASCII 값은 대문자만 인식, 'a' ➔ 'A', 'z'➔'Z'
  - GetKeyboardState()
    - 256 virtual Key의 상태를 256-byte 버퍼에 복사
  - SetKeyboardState()
    - 256-byte의 배열 값을 키보드 상태로 복사 ➔ 키보드 상태 초기화에 사용

- Mouse
  - GetCursorPos()
    - 바탕 화면 기준으로한 마우스의 위치
  - ScreenToClient()
    - 위치를 해당 윈도우 작업 영역의 상대적인 위치로 변경

  - WM_MOUSEWHEEL: 마우스의 휠의 변경 값은 윈도우 메시지 WM_MOUSEWHEEL
  - 에서 처리. _WIN32_WINNT 값을 최소 0x0400 이상 windows.h 파일을 인클루드 하기 전에 설정해야 사용 가능

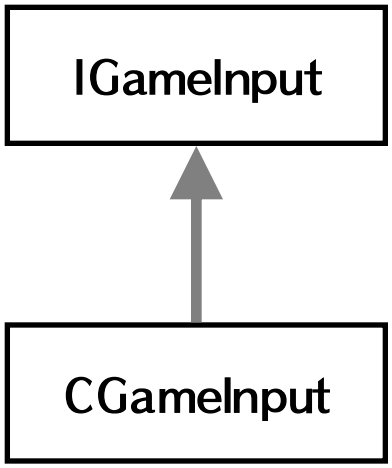
Ex)

```
#define _WIN32_WINNT 0x0400
#include <windows.h>
```

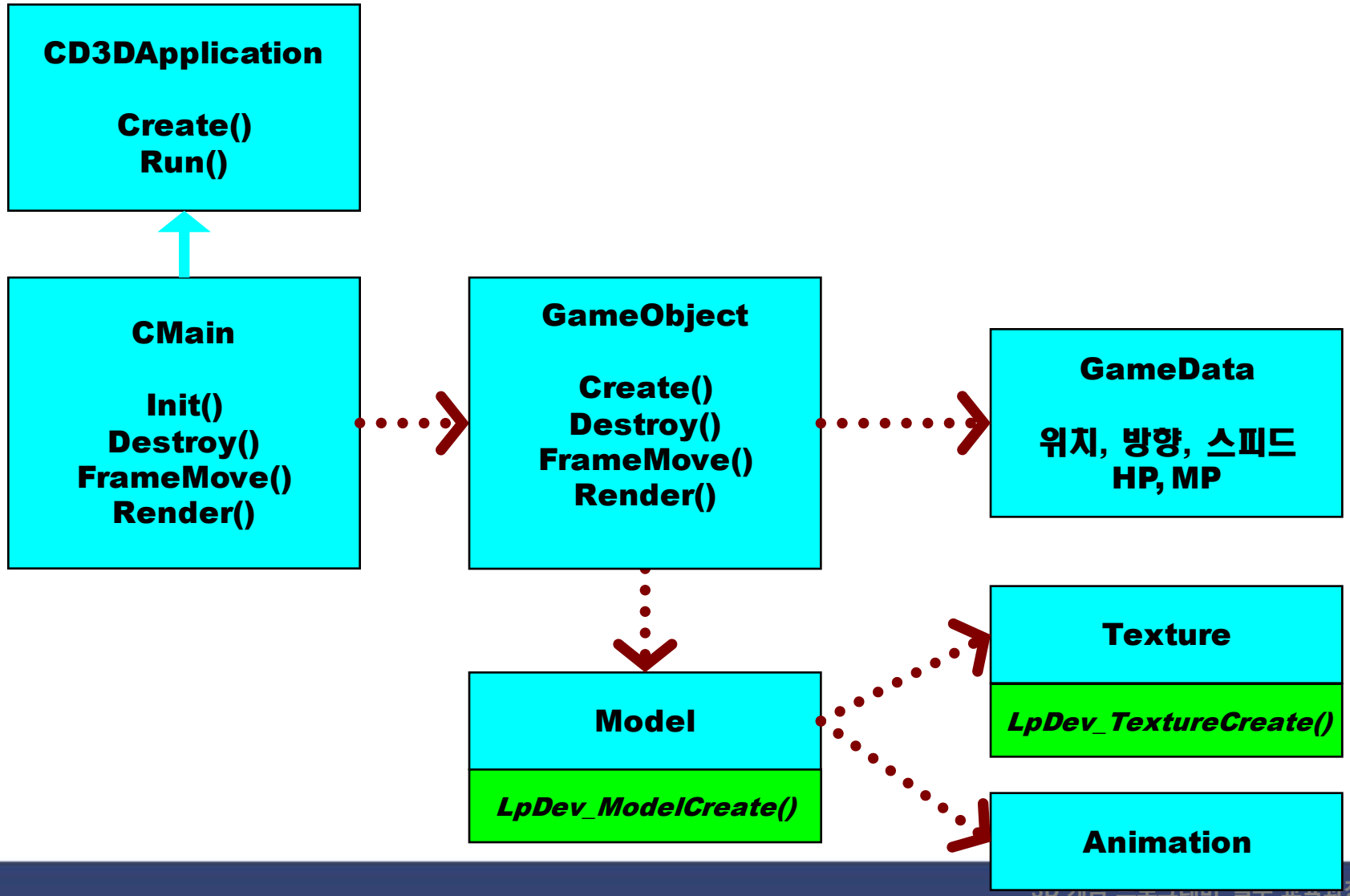
- GameInput class
  - Game Data: 온라인게임을 만들었을 때 서버와 주고 받는정보
  - GameModel: 렌더링 데이터. 서버와 무관한 정보

Ex)

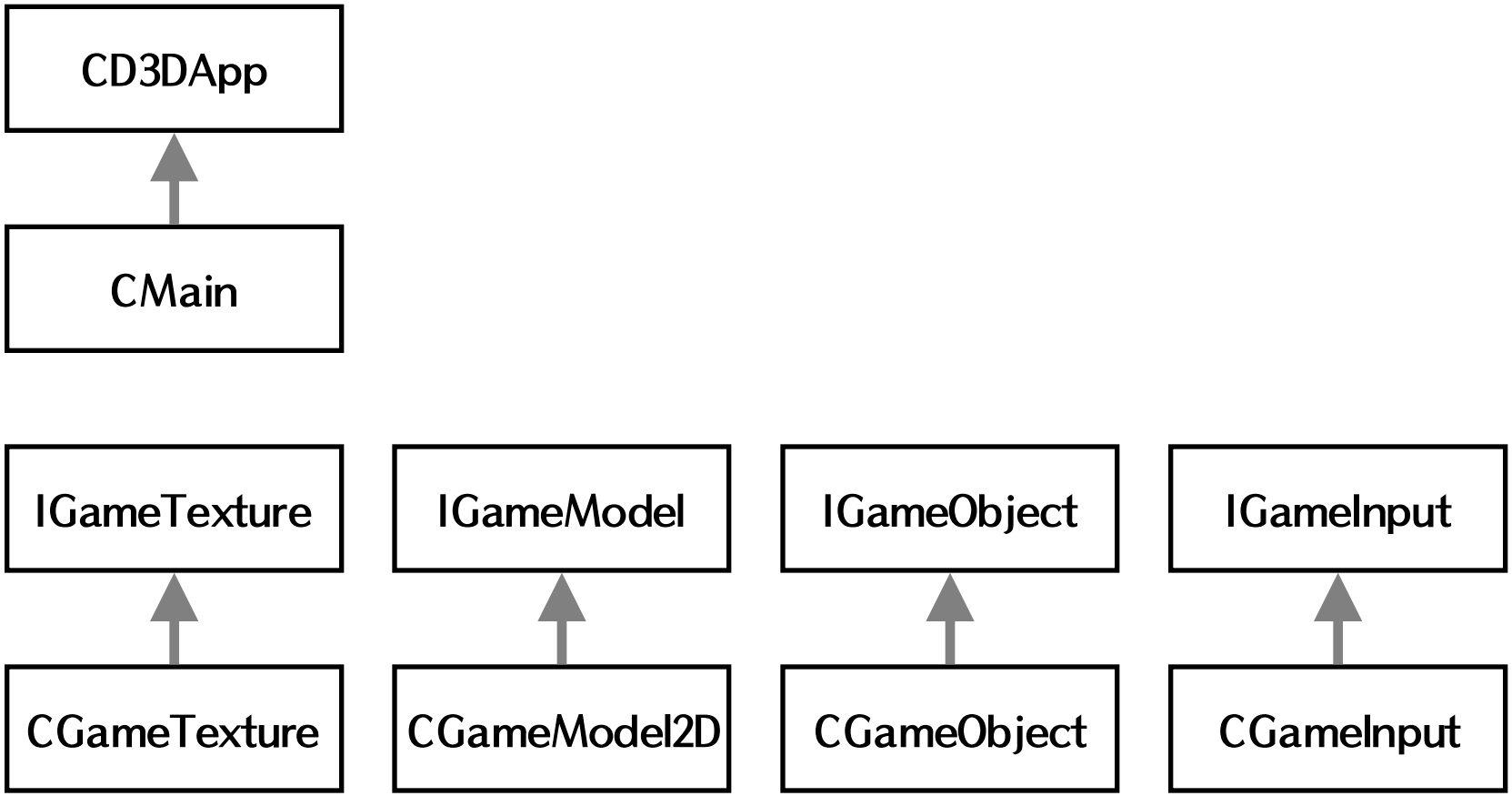
```
interface IGameInput
{
	virtual ~IGameInput(){};

	virtual int		Create() = 0;
	virtual void		Destroy() = 0;
	virtual int		FrameMove() = 0;
	virtual LRESULT	MsgProc(HWND,UINT, WPARAM, LPARAM)=0;

public:
	 virtual BOOL	Key{Down|Up|Press|State}(INT)=0;
	 virtual BOOL	Btn{Down|Up|Press|State}(INT)=0;

	const float* GetMousePos()=0;
	const float* GetMouseEps() =0;
};


class CGameInput : public IGameInput
{
protected:
	HWND		m_hWnd;

	BYTE		m_KeyCur[MAX_INPUT_KEY];   // 키보드 현재 상태
	BYTE		m_KeyOld[MAX_INPUT_KEY];   // 키보드 이전 상태
	BYTE		m_KeyMap[MAX_INPUT_KEY]; // 키보드 맵
public:
	...
};
```



- 게임 클래스 추상화 ➔



- 게임 클래스 추상화 ➔



- C 형식으로 작성된 코드를 클래스로 전환하시오. 클래스 이름은 CD3DApplication으로 합니다.

- CD3DApplication 클래스를 상속받는 CMain 클래스를 만들고 CD3DApplication 클래스에는 윈도우와 Direct3D와 관련된 코드만 남기고 나머지는 CMain 클래스로 옮기시오.
  - Init(), Destroy(), FrameMove(), Render() 함수는 CD3DApplication에서 가상함수로 정하고, CMain에서도 구현하시오.

- 텍스처 클래스, 모델 클래스, 오브젝트 클래스, 인풋 클래스를 각각 만들고 이를 추상화 하시오.