15. 다양한 타입의 오브젝트와 플레이어 클래스

```h
#pragma once

#include "Game.h"

class CRef
{
protected:
	CRef();
	virtual ~CRef() = 0;

protected:
	int		m_iRef;

public:
	void AddRef()
	{
		++m_iRef;
	}

	int Release()
	{
		--m_iRef;

		if( m_iRef == 0 )
		{
			delete this;
			return 0;
		}
		return m_iRef;
	}
};
```

```cpp
#include "Ref.h"

CRef::CRef()
	:
	m_iRef( 1 )
{
}

CRef::~CRef()
{
}
```

CRef 클래스는 이미지 같은 자원을 담아둘 것임.
이를 여러 오브젝트들이 공유할 수 있도록 내부에 레퍼런스 카운트를 만들어줌.

```h
#pragma once

#include "../Game.h"
#include "../Ref.h"

class CObject : public CRef
{
protected:
	CObject();
	CObject( const CObject& obj );
	virtual ~CObject();

protected:
	class CScene* m_pScene;
	class CLayer* m_pLayer;

protected:
	string		m_strTag;
	POSITION	m_tPos;
	_SIZE		m_tSize;
	POSITION	m_tPivot;

public:
	CScene* GetScene() const
	{
		return m_pScene;
	}
	CLayer* GetLayer() const
	{
		return m_pLayer;
	}
	string GetTag() const
	{
		return m_strTag;
	}

	POSITION GetPos() const
	{
		return m_tPos;
	}

	_SIZE GetSize() const
	{
		return m_tSize;
	}
```

자신이 속한 씬과 레이어

```h
public:
	// scene
	void SetScene( class CScene* pScene )
	{
		m_pScene = pScene;
	}
	// layer
	void SetLayer( class CLayer* pLayer )
	{
		m_pLayer = pLayer;
	}
	// tag
	void SetTag( const string& strTag )
	{
		m_strTag = strTag;
	}
	// pos
	void SetPos( const POSITION& tPos )
	{
		m_tPos = tPos;
	}
	void SetPos( const POINT& pos )
	{
		m_tPos = pos;
	}
	void SetPos( float x, float y )
	{
		m_tPos.x = x;
		m_tPos.y = y;
	}
	// size
	void SetSize( const POSITION& tSize )
	{
		m_tSize = tSize;
	}
	void SetSize( const POINT& size )
	{
		m_tSize = size;
	}
	void SetSize( float x, float y )
	{
		m_tSize.x = x;
		m_tSize.y = y;
	}

public:
	virtual bool Init() = 0;
	virtual void Input( float fDeltaTime );
	virtual int Update( float fDeltaTime );
	virtual int LateUpdate( float fDeltaTime );
	virtual void Collision( float fDeltaTime );
	virtual void Render( HDC hDC, float fDeltaTime );
```

```h
public:
	template <typename T>
	static T* CreateObj( const string& strTag, class CLayer* pLayer = nullptr )
	{
		T* pObj = new T;

		if( !pObj->Init() )
		{
			SAFE_RELEASE( pObj );
			return nullptr;
		}

		if( pLayer )
		{
			pLayer->AddObject( pObj );
		}

		pObj->AddRef();

		return pObj;
	}
};
```

오브젝트 생성

Object 클래스는 CRef 클래스를 상속받음.
때문에 기존에 Object 클래스에 있던 레퍼런스 카운트 부분을 삭제함.

MoveObj

```h
#pragma once

#include "Object.h"

class CMoveObject : public CObject
{
protected:
	CMoveObject();
	CMoveObject( const CMoveObject& obj );
	virtual ~CMoveObject();

public:
	virtual bool Init() = 0;
	virtual void Input( float fDeltaTime );
	virtual int Update( float fDeltaTime );
	virtual int LateUpdate( float fDeltaTime );
	virtual void Collision( float fDeltaTime );
	virtual void Render( HDC hDC, float fDeltaTime );
};
```

StaticObj

```h
#pragma once

#include "Object.h"

class CStaticObject : public CObject
{
protected:
	CStaticObject();
	CStaticObject( const CStaticObject& obj );
	virtual ~CStaticObject();

public:
	virtual bool Init() = 0;
	virtual void Input( float fDeltaTime );
	virtual int Update( float fDeltaTime );
	virtual int LateUpdate( float fDeltaTime );
	virtual void Collision( float fDeltaTime );
	virtual void Render( HDC hDC, float fDeltaTime );
};
```

Move/Static 오브젝트는 Object 클래스 상속.

```h
CMoveObject::CMoveObject( const CMoveObject& obj )
	:
	CObject( obj )
{
}
```

이런식으로 복사 생성자 경우 부모 복사생성자 호출해줌.

player는 MoveObject 상속받음

Player

```h
#pragma once

#include "MoveObject.h"

class CPlayer : public CMoveObject
{
private:
	CPlayer();
	CPlayer( const CPlayer& player );
	~CPlayer();

public:
	virtual bool Init();
	virtual void Input( float fDeltaTime );
	virtual int Update( float fDeltaTime );
	virtual int LateUpdate( float fDeltaTime );
	virtual void Collision( float fDeltaTime );
	virtual void Render( HDC hDC, float fDeltaTime );
};
```