14. 레퍼런스 카운트와 상속구조

예를 들어 플레이어 피격 UI가 있다고 해보자. UI 호출시 모반객트들을 하나 하나 검사하는 것보다 플레이어 객체가 UI를 들고 있고 피겨시해당 UI를 접근/호를 하는 것이 효율적이다.

그런데, 해당 피격 UI가 1면되어 사라졌을 경우 플레이어는 이 사실을 모르고 계속 해당 메오리 공간을 참조하게 되어 문제가 반생한다.

객체하나를 여러 객체들이 공유할 때 발생하는 객체 공유의 문제.

→ 객체들에게 자신을 참조하는 횟수를 알고 있게 함.

이를 레퍼런스 카운트 라고 하는다. * shared-ptround 사용

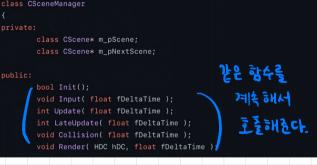
```
CScene::~CScene()
{
    Safe_Delete_VecList( m_LayerList );
}
```

소명자를 토흘하는 것이 아닌 레피언스 카운트를 줄이는 release를 토흘해주고, 오브젝트 객체가 자기자신의 tef count가 이미되면 소명자를 스스로 토흘라다.

•상속구조

```
### Addition of the content of the c
```

기상위 관리자 Core가 hdc 만들어 센 메니저의 하수들을 즐글.
class CScene* m_pSc class CScene* m_pNc



₩ 호출
 레이어 호출
 레이어의 각 모브젝트
 호출