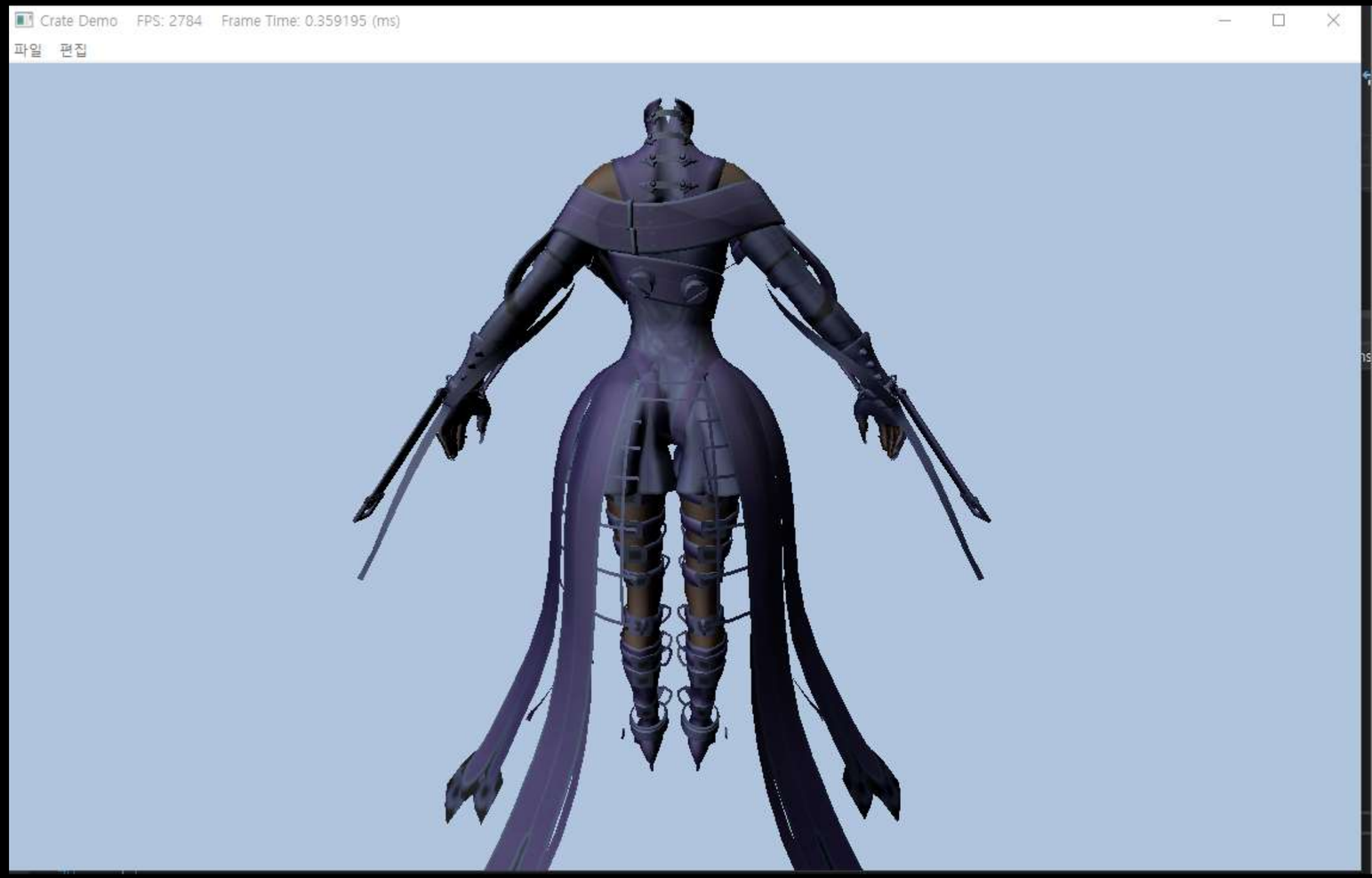
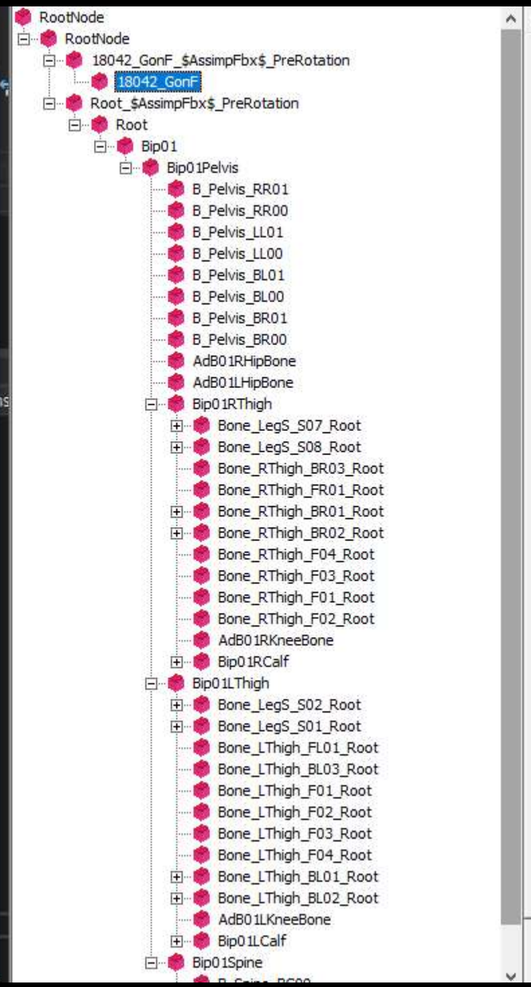
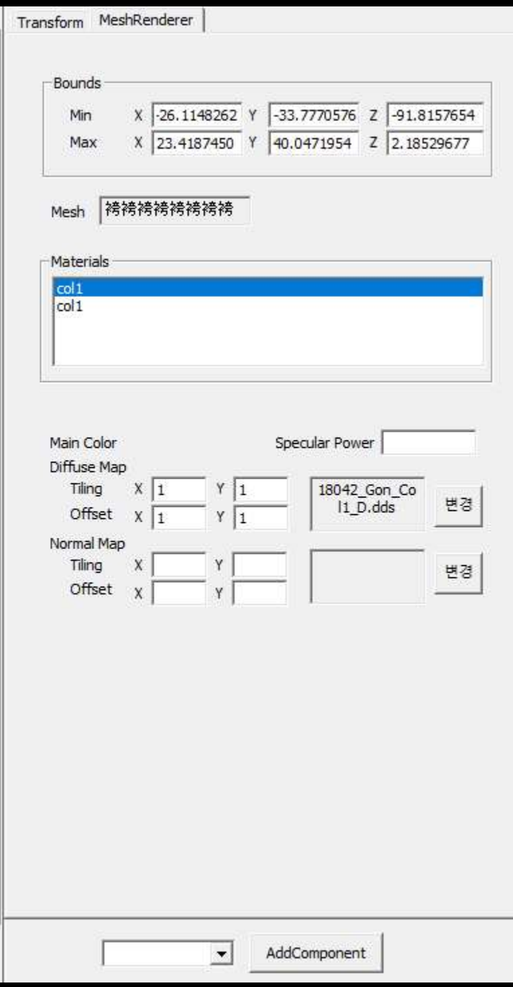
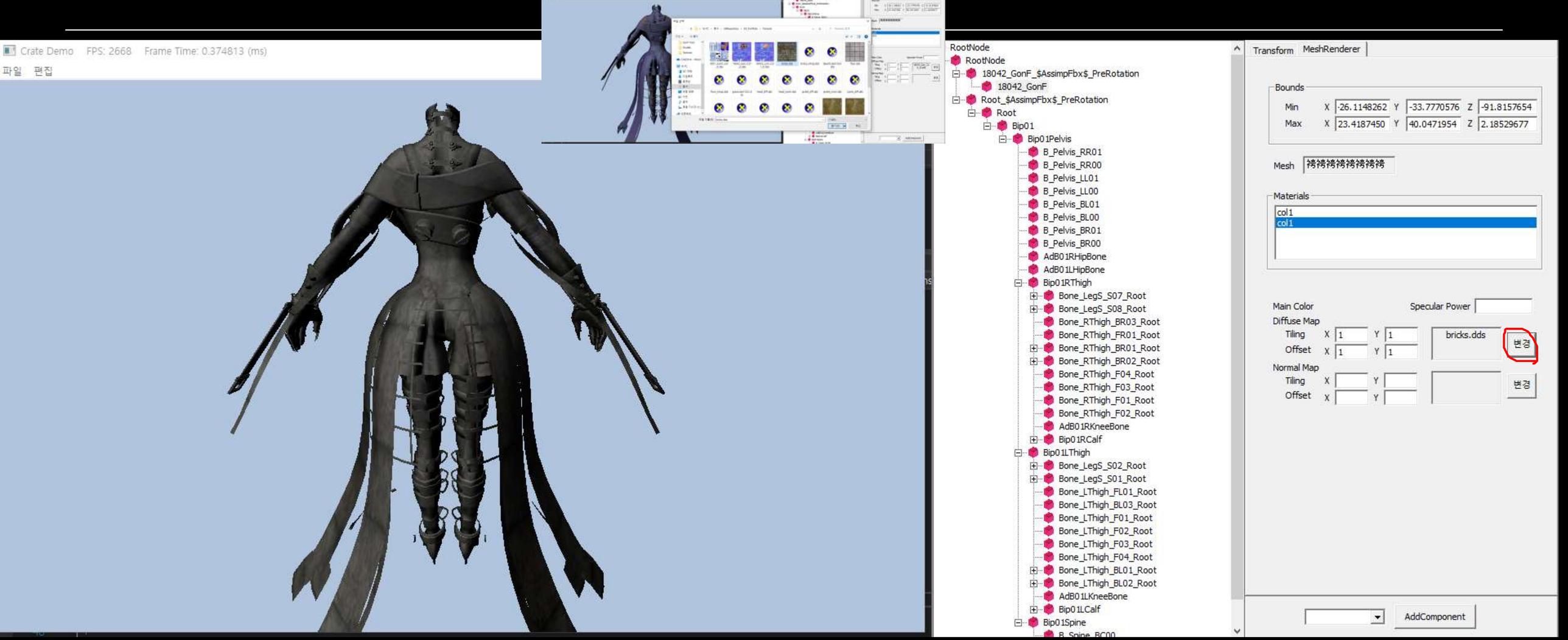
장진성_포트폴리오



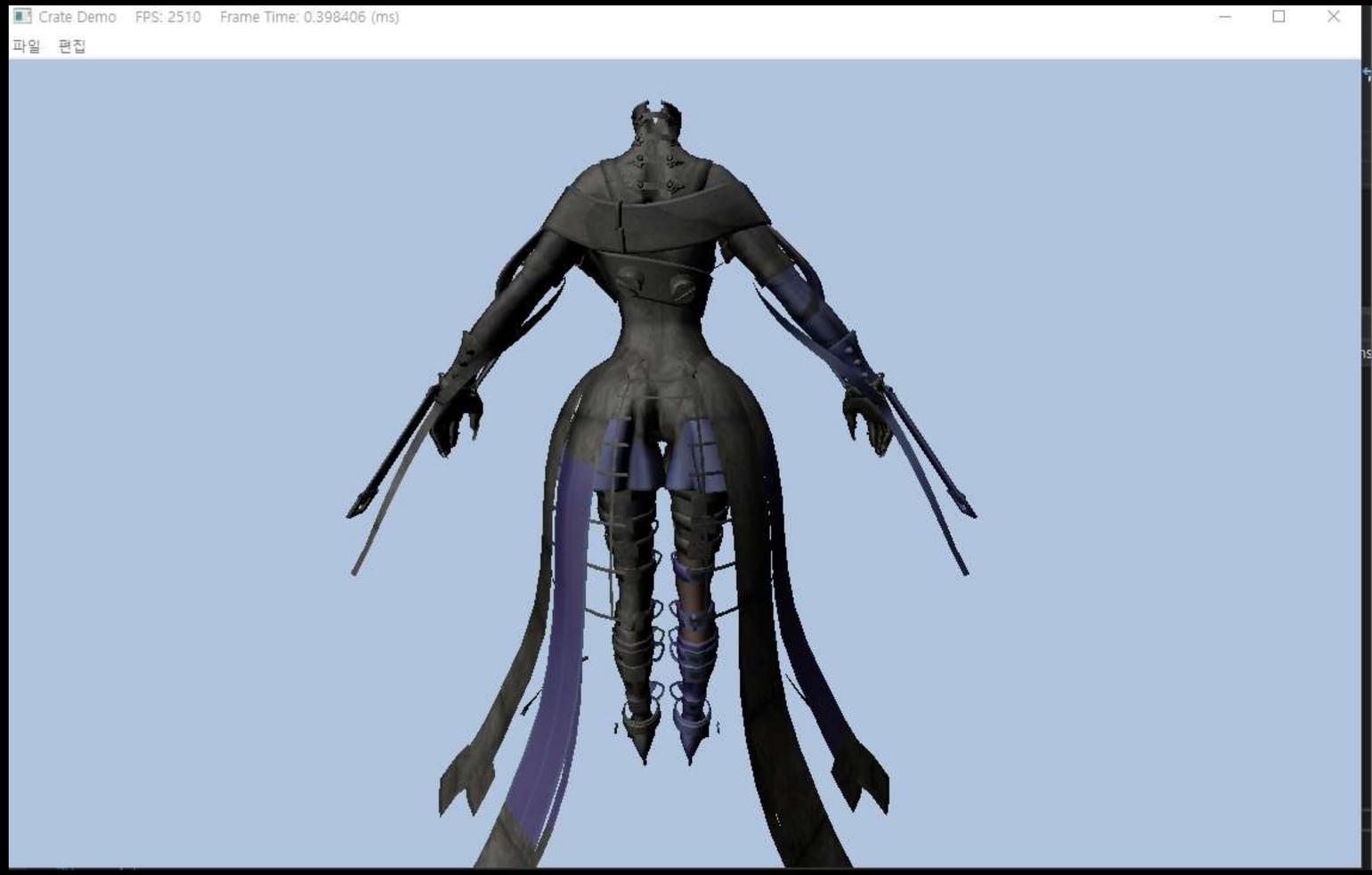


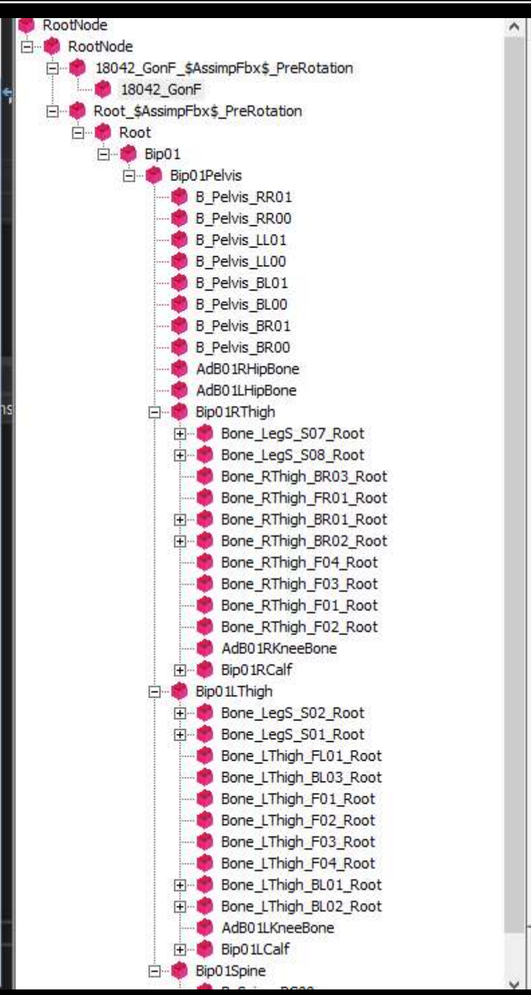


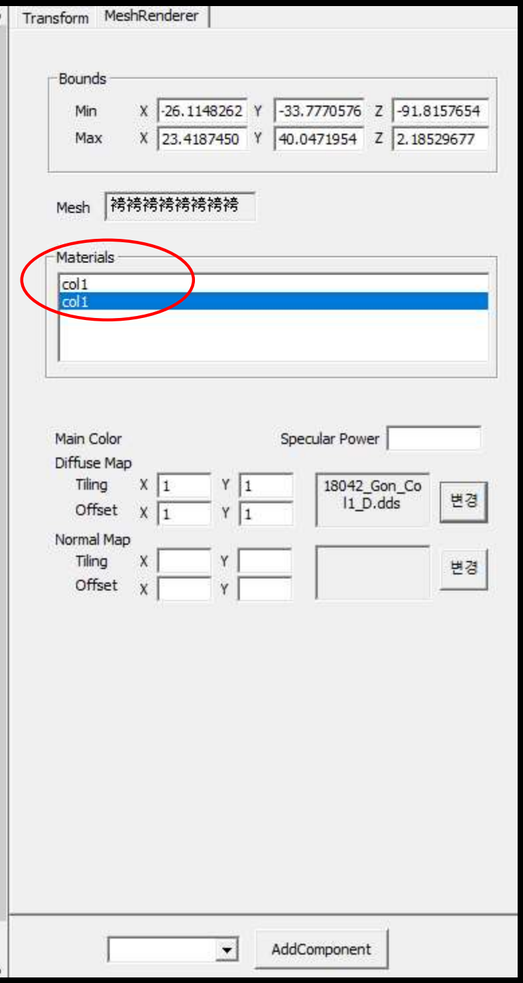
- Assimp Library로 fbx 파일을 읽어와 렌더링 합니다.
- 오브젝트는 컴포넌트의 집합으로 정의됩니다. 오브젝트는 Transform 컴포넌트를 기본으로 가지고 다른 컴포넌트들을 추가 할 수 있습니다.
- 새로운 기능을 구현할 때 Component 클래스를 상속한 클래스를 만들고, 추가하면 작동하도록 설계하였습니다.
- 렌더링은 MeshRenderer 컴포넌트를 오브젝트에 추가하여 실행됩니다.



- 텍스쳐 파일을 불러와 Diffuse Map을 변경 할 수 있습니다.
- 명령패턴을 이용해 실행되며 Accelrator 단축키 설정을 이용해 Ctrl+z를 누르면 명령취소가 되게 구현했습니다.







- 하나의 Mesh는 여러개의 Material로 구성될 수 있고 Material 별로 Diffuse Map을 변경 할 수 있습니다.

파란색은 클래스명 / 검정색은 멤버변수 / 초록색은 멤버함수 입니다.

- 프로그램 설명

1. 기본적인 렌더링, 업데이트와 같은 게임로직은 Scene 클래스에서 이루어집니다.

2. Scene에 있는 오브젝트들을 렌더링, 업데이트 합니다.

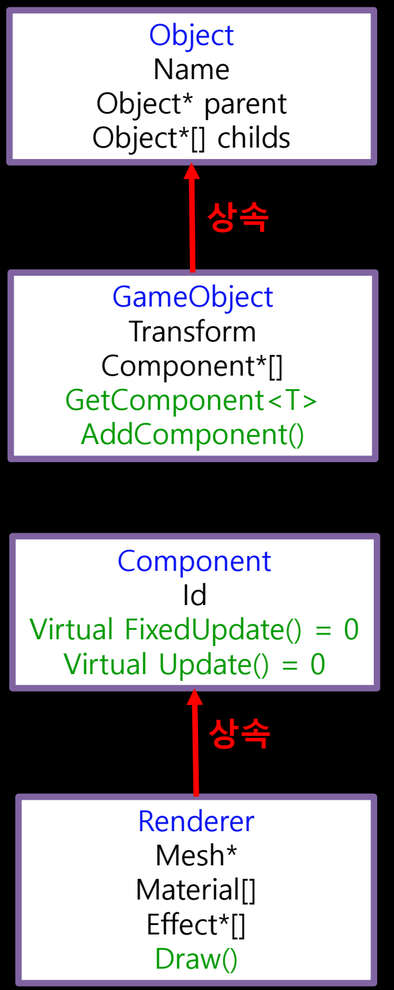
3. 오브젝트는 컴포넌트의 집합으로 이루어집니다.

4. 컴포넌트 매니저에서 모든 컴포넌트를 관리합니다.

5. 컴포넌트 매니저의 모든 컴포넌트를 업데이트, 렌더링 합니다.

렌더링 구조

주요 클래스



- 중점적으로 생각했던 부분

Object는 Component의 포인터만을
가지고 있고, 실제 Component는
Manager에서 벡터컨테이너로 관리해
Cache Hit를 높혔습니다.

```cpp
template<typename compType>
inline Component* ComponentMgr::SwapEnable(std::vector<compType>& vec, int & enableCount, int idx)
{
    //비활성화 컴포넌트인지 검사
    assert(idx >= enableCount);


    //비활성화된 컴포넌트를 제일 앞에 있는 비활성화된 컴포넌트와 바꿈
    std::swap(vec[enableCount], vec[idx]);

    //id와 index를 매핑하는 해쉬맵 업데이트
    idMap[vec[enableCount].id] = enableCount;
    idMap[vec[idx].id] = idx;

    //활성화된 카운트 수 증가
    enableCount++;

    return &vec[enableCount - 1];
}
```

Component를 활성화 시키는 함수입니다.
활성화된 컴포넌트의 개수보다 인덱스가 작으면 활성화입니다.
항상 앞쪽에 활성화된 컴포넌트를 모아두고,
렌더링이나 업데이트시 활성화된 앞쪽만 동작합니다.

```cpp
class ComponentMgr
{
private:
    std::vector<MeshRenderer> meshRenderers;
    std::vector<SkinnedMeshRenderer> skinnedMeshRenderers;

    //component의 id와 배열 index 매핑
    std::unordered_map<std::string, int> idMap;
    //component의 type 매핑
    std::unordered_map<std::string, ComponentType> typeMap;

private:
    //Component를 만들때 사용할 id넘버
    int creatingIdNum;
    //활성화 된 컴포넌트의 개수
    int enableCount_meshRenderer;
    int enableCount_skinnedMeshRenderer;
```

Component Manager는 각 컴포넌트를 vector로 관리합니다.
각 컴포넌트마다 활성화된 컴포넌트의 개수를 가지고
있습니다.