**황인원 Email: inwon481@gmail.com**

1. **Branch XFile 테스트**

: X파일을 읽어올 때 X파일 안에 애니메이션이 있는지 없는지에 따라, 리소소를 분류하여 저장하기 위하여 테스트하기.

**에러**

: CXFileMgr클래스 에서 X파일을 만든다음 만든 X파일로 CreateEnumObject함수를 호출했는데 S\_FALSE값 리턴.

\_pXFile->CreateEnumObject((LPVOID)L"tiger.x", D3DXF\_FILELOAD\_FROMFILE, &\_pEnum)))

\_pXFile->CreateEnumObject((LPVOID)"tiger.x", D3DXF\_FILELOAD\_FROMFILE, &\_pEnum)))

**해결**

* 파일명이 잘못됨.

**결과**

**:** 각 x파일 별로 Template확인 및 각 X파일 내의 데이터를 파싱하여 원하는 리소스를 뺴낼 수 있다는 것을 확인함.

**고려사항**

1. ID3DXFile을 보관해야 할 필요가 있나?
   1. X파일을 읽어서 X파일 안에 있는 파싱해서 각 리소스 컨테이너에 넣어준다.
   2. 나중에 변경된 리소스를 X파일로 저장할 일이 생긴다.
2. 결론
   1. A만 고려하면 없어도 되지만 B를 생각하면 필요함.
   2. 유니티 엔진에 스크립트로 프리팹을 생성하는 기능이 B기능이라고 생각됨.

**진행해야할 상황**

1. x파일을 관리하는 클래스 만들기
   1. 디렉토리에 있는 X파일 모두 불러오기.
   2. map<string name, ID3DXFile \*>에 저장.
2. 파싱 하는 클래스 만들기.
   1. MeshContainer 부터 차근차근 만들기
3. 리소스 매니저 만들기.
   1. MeshContainer 부터 차근차근 만들기.
4. **XFile 관리하는 클래스 만들기**

WBCS MBCS간 변환

<http://milkkal.tistory.com/14>

std::map 사용법

<http://piml.tistory.com/entry/STLmap-%EC%82%AC%EC%9A%A9%EB%B0%A9%EB%B2%95>

**고려사항**

1. LPD3DXFILE 과 LPD3DXFILEENUMOBJECT를 private 영역의 자료구조에 넣었다. 그런데 public 함수를 통해 LPD3DXFILE 과 LPD3DXFILEENUMOBJECT에 접근하는데 접근해서 Release를 해버릴 수도 있다.
   1. Public 함수를 private로 변경 후 friend 클래스 내에서만 사용해 원하는 클래스 안에서만 접근 하게 한다.
   2. LPD3DXFILE 과 LPD3DXFILEENUMOBJECT를 직접 접근하게 하지 말고 함수를 통해 접근하게 한다.
   3. A, B중 어떤 방법을 선택할 것인가? 둘 다 쓸 것인가?

**진행해야할 상황**

1. 1-0 의 고려상황 때문에 X파일을 관리 해야될 것이라고 생각했지만, 실제 X파일 데이터를 저장하거나 불러오는 것은 D3DXFILE의 CreaeteEnumObject, CreateSaveObject에 의해 이루어지는 것을 확인.
   1. XFileManager을 없애버린다. D3DXFILE 객체는 몇 개 필요할까?
      1. 로딩하는 과정이 쓰레드에 의해 이루어질 가능 성이 있는가? OK
      2. 저장하는 과정이 쓰레드에 의해 이루어질 가능 성이 있는가? Almost NO
   2. D3DXFILE이 하는 일은 템플릿 등록, CreateEnumObject, CreateSaveObject 1개만 생성해서 동기화 시켜주는 주는 방법.
   3. 로딩하거나 저장할때마다 D3DXFILE 생성하는 방법.
2. 1항목의 B, C중 어느 것 이 더 효율적인가? 모름 일단 1-B방법으로 진행
3. **파싱**

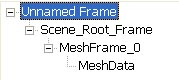
**에러**

: 프로그램 실행 종료 할때마다 LNK1168 .exe파일을 쓰기용으로 열 수 없다는 에러가 뜸.

프로그램 종료할 때 WM\_QUIT을 보내서 종료하는게 아니라 bool값을 이용해서 종료했기 때문에 정상적으로 종료가 되지 않았음. 이후 LNK1168에러는 발생하지 않음.

**X파일구조**

1. Mesh
   1. 단일 메쉬
2. Frame
   1. 프레임 서브 메쉬가 존재



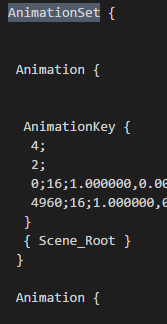
1. Frame + AnimationSet
   1. 애니메이션과 스키닝이된 메시

* 각 메쉬나, 프레임, 애니메이션 셋에는 서브데이터로 머테리얼리스트, 텍스쳐, 애니메이션 등등이 있음.

**TID\_D3DRMMesh타입의 D3DXFILEDATA를 파싱했을 때 얻을 수 있는 것들.**

1. 메쉬 (안에 버텍스, 인덱스 등의 정보가 있다.)
2. 머테리얼버퍼
   1. 버퍼를 통해 머테리얼들을 얻어낼 수 있음.
3. 텍스쳐(텍스쳐 파일 이름, UV정보 등)
   1. 텍스쳐이름을 통해 텍스쳐를 로드할 수 있음.
4. 데이터에 스킨정보가 있으면 얻을 수 있음.

**X파일 애니메이션 정보 구조**



애니메이션Key의 첫번째 정보는 키타입을 나타냄.

* 키타입 == 1 : 애니메이션키는 포지션값을 나타냄.
* 키타입 == 2 : 애니메이션키는 스케일값을 나타냄.
* 키타입 == 3 : 애니메이션키은 로테이션값을 나타냄.
* 키타입 == 4 : 애니메이션키는 매트릭스값을 나타냄.

2번째 정보는 키의 개수를 나타냄.

3번째정보는 키타입에따른 정보(시간, 정보개수, (w,x,y,z)값)

**완료**

* 메쉬파서 작성(CMeshParser, CSkinnedMeshParser)

**진행해야할 사항**

* Bone파싱하기.
* AnimationKey얻어내기;(파싱)
* AnimationKey를 애니메이션에 넣기(애니메이션 클래스 만들기)
* AnimationController에 애니매이션 넣기(애니메이션 컨드롤러 만들기)