원화 외 작업물

프로그래밍, TA, 3D 모델링

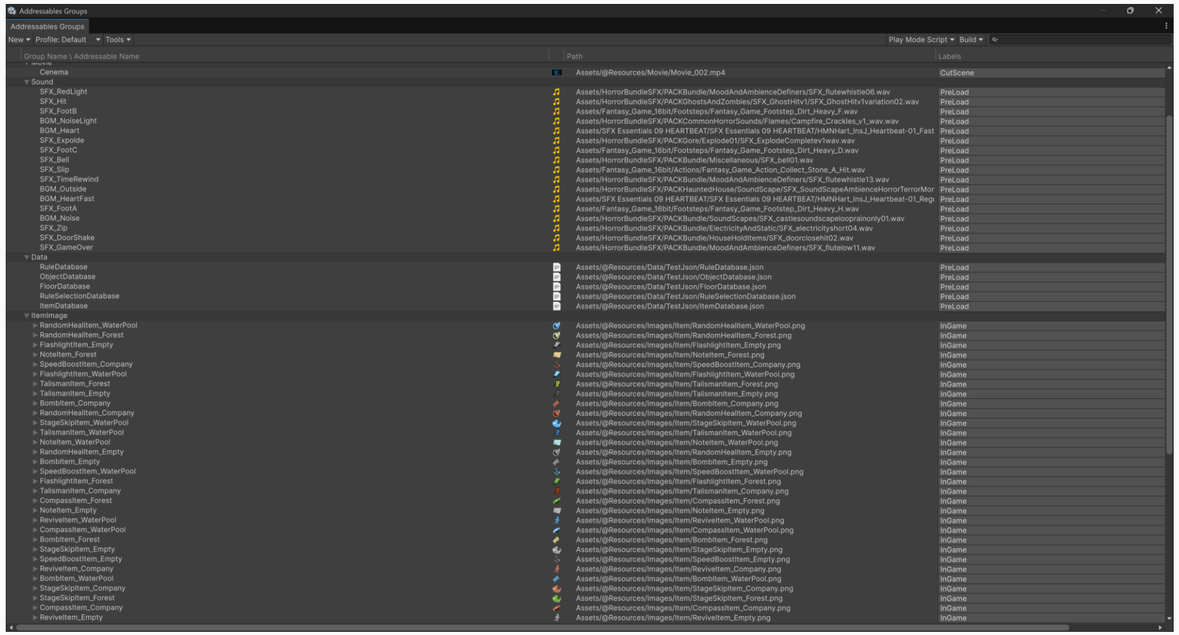
프로그래밍: 데이터 기반 설계

```
BaseObject SpawnObjectByIndex(int index){
    var d = Managers.Data.ObjectDic[index];

    ObjectData data;

    data = new ObjectData();
    data.TickScinceCalled = 0;
    data.MoveableThrewWall = d.MoveableThrewWall;
    data.IsBinded = false;
    data.ObjectType = d.ObjectType;
    data.BaseObjectType = GetBaseType(data.ObjectType);
    data.Damage = d.Damage;
    data.MoveInterval = d.MoveInterval;
    data.RotationDir = Vector2Int.zero;
    data.TempValue1 = d.TempValue1;
    data.TempValue2 = d.TempValue2;
    data.TempValue3 = d.TempValue3;
    data.IsDestroyThis = false;

    return SpawnObject (d.ObjectType, data);
}
```

모든 게임 규칙, 아이템 스탯, 오브젝트 능력치를 csv파일로 외부화하여 코드 수정 없이 데이터만 바꿔서 게임 밸런스와 콘텐츠를 수정

DataManager가 시작 시 JSON을 파싱해 Dictionary로 캐싱하고, 런타임에 RuleBook, ItemInventory 등이 ID로 데이터를 조회하여 동적으로 게임 오브젝트를 생성

프로그래밍: Addressables 리소스 관리

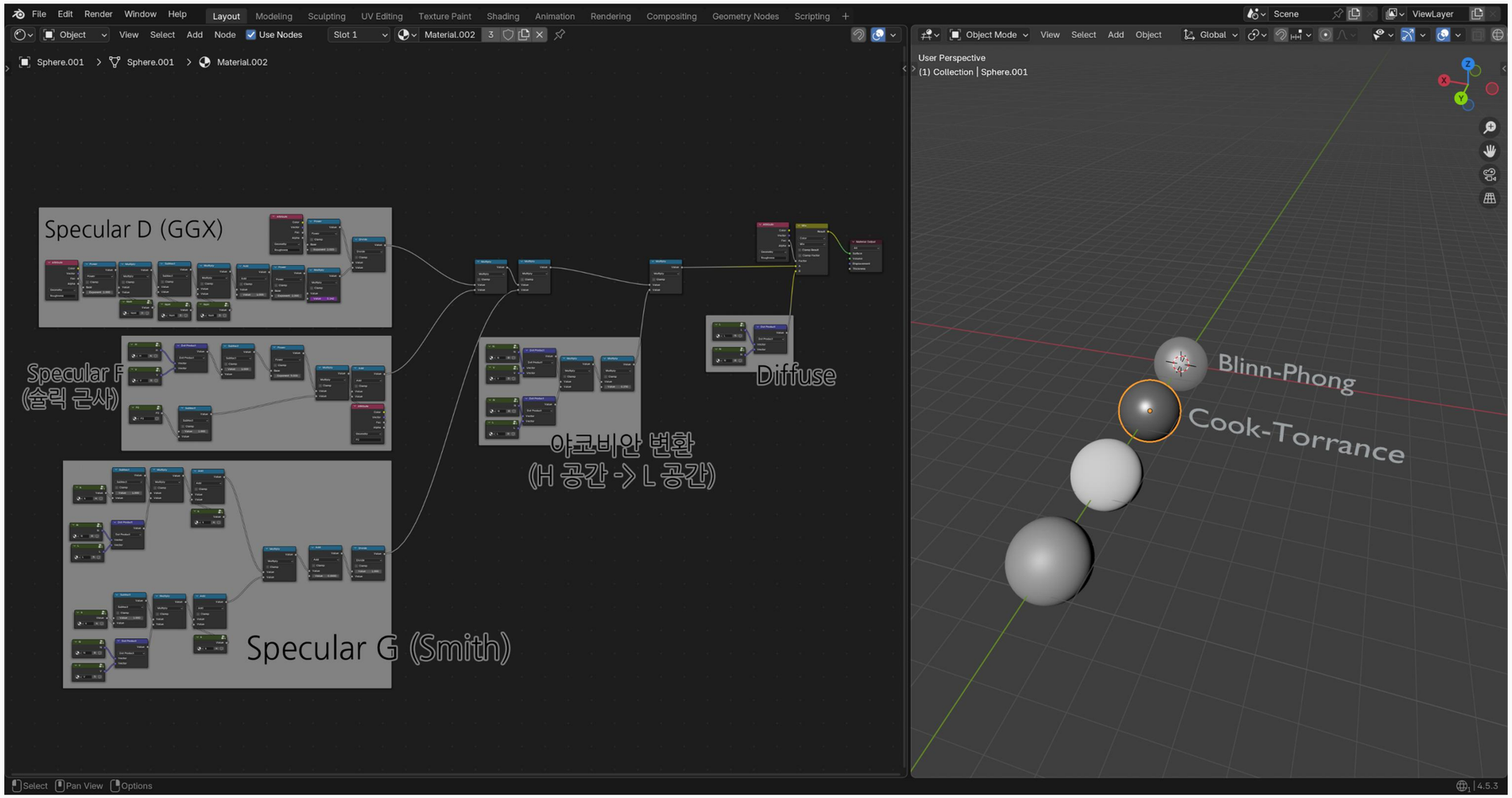


PreLoad, InGame, CutScene 레이블로 리소스를 분리하여 씬별로 필요한 에셋만 비동기 로딩

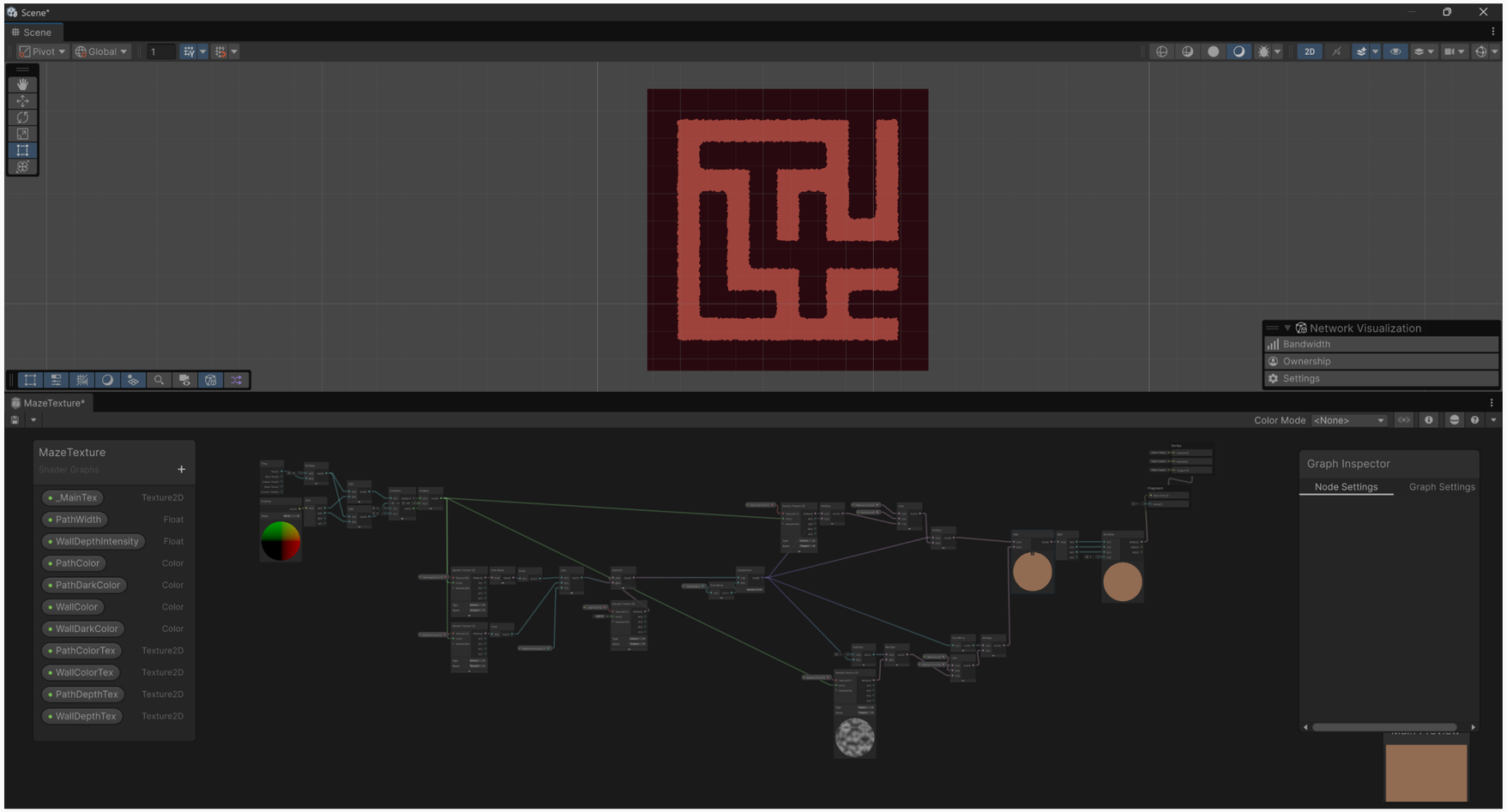
Dictionary 기반 캐싱으로 동일 리소스 중복 로드를 방지

모든 리소스를 Addressables + .csv 로 관리하여, 유니티 인스펙터를 통하지 않고도 다양한 컨텐츠 추가 및 수정 가능

TA: Shader

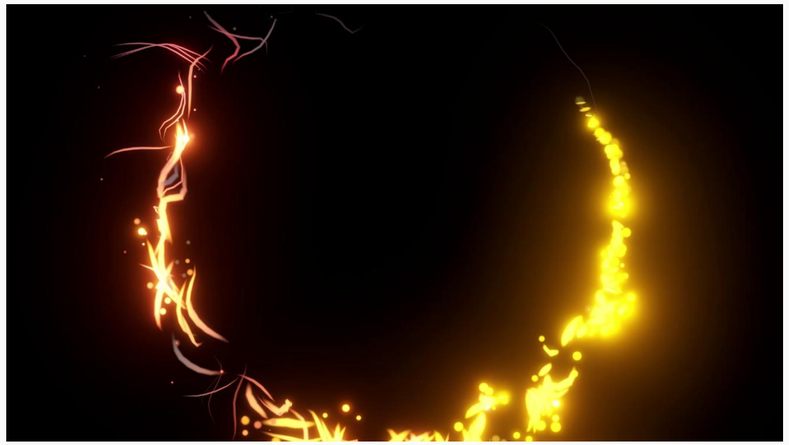

Specular D (GGX)
Specular F
(슐릭 근사)
Specular G (Smith)
아코비안 변환
(H 공간 -> L 공간)
Diffuse
Blinn-Phong
Cook-Torrance

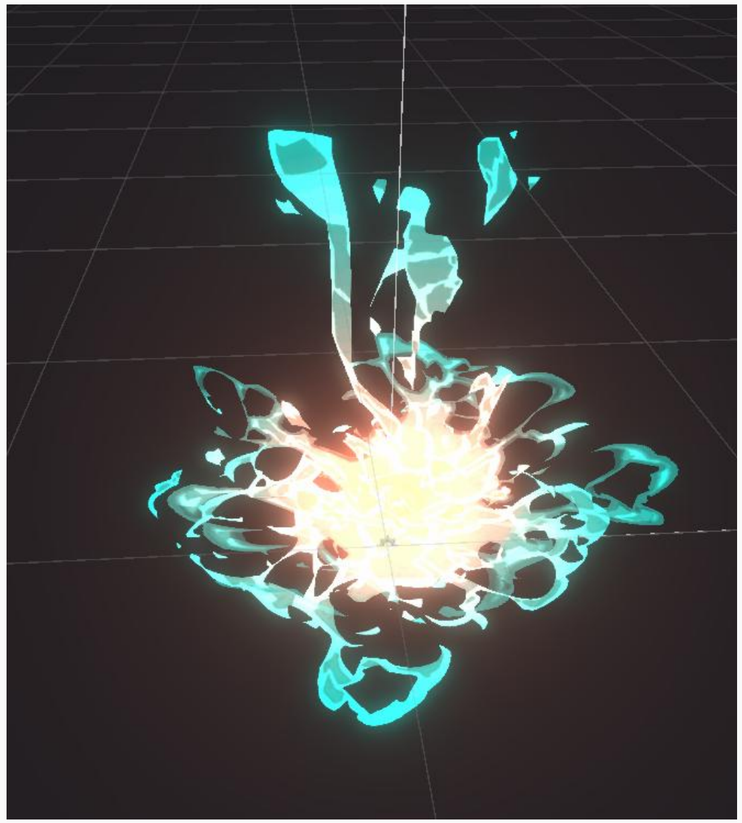
TA: Procedural Material



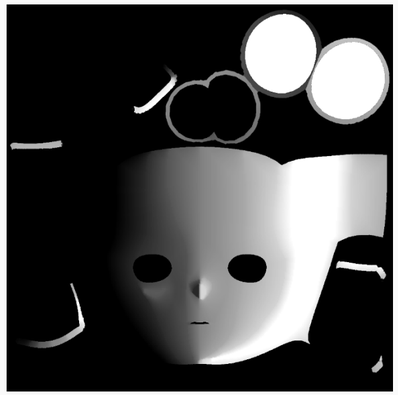
게임 'MOTA' 에서 미로를 타일맵으로 생성한 이후 material 을 통해 텍스처를 입힘

TA: 간단한 수준의 이펙트





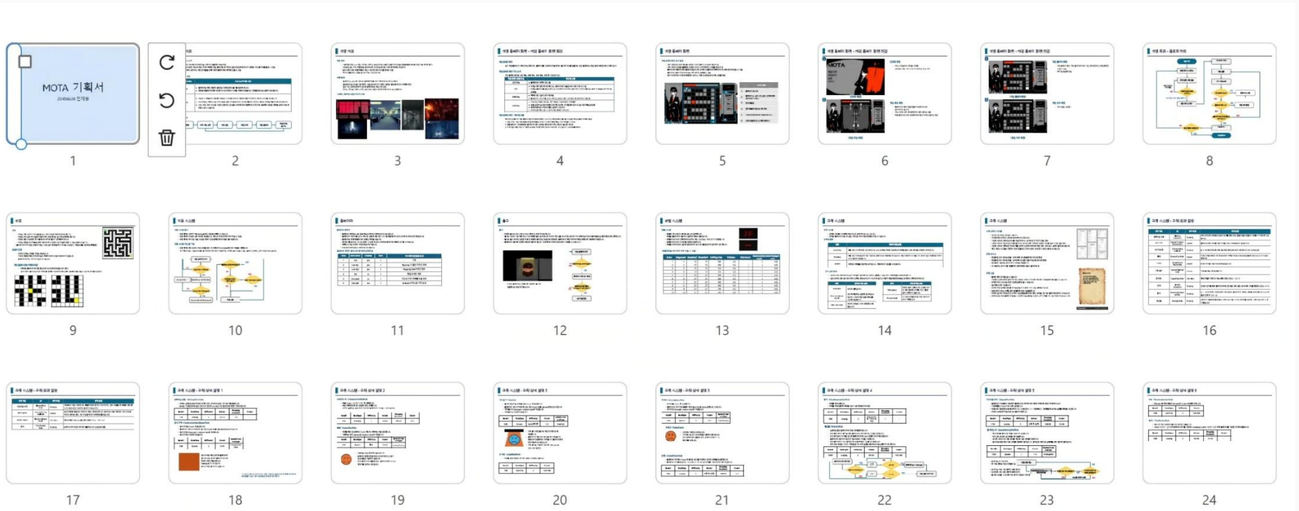
3D 모델링: 캐릭터



SDF 이미지

그 외

UI



간단한 문서 작성