# **던전 마스터: 폴더 구조 및 규칙 가이드 v1.9**

## **1. 개요**

본 문서는 "던전 마스터" 프로젝트의 일관성과 효율성을 유지하기 위한 공식 폴더 구조와 에셋 관리 규칙을 정의합니다. 모든 팀원은 이 가이드를 숙지하고 준수하여, 기술 부채를 최소화하고 원활한 협업 환경을 구축해야 합니다.

이 구조의 핵심 철학은 **기능 기반(Feature-Based) 아키텍처**입니다. 이는 각 기능이 독립적인 모듈처럼 작동하도록 하여, 개발의 예측 가능성을 높이고 유지보수 비용을 절감하는 것을 목표로 합니다.

**v1.9 업데이트 내역 (최종 완성):**

* **Feature별 Art/Audio 규칙 표준화:** 모든 기능 폴더는 필요 시 동일한 위치에 Art/Audio 폴더를 가질 수 있음을 명시하고 예시에 반영.
* **Prefab 하위 구조 규칙 표준화:** Prefab 폴더의 하위 분류 패턴을 [유형/역할]로 통일하여 명시.
* **Addressables 관리 규칙 보강:** Unity 자동 생성 관련 주의 문구 추가.

## **2. 폴더 구조 설계도 (Overall Structure)**

Assets/  
├─ \_Project/  
│ ├─ Features/  
│ │ ├─ Core/  
│ │ ├─ Dungeon/  
│ │ ├─ Combat/  
│ │ ├─ Units/  
│ │ ├─ InventorySystem/  
│ │ ├─ Equipment/  
│ │ ├─ MetaGame/  
│ │ └─ UI/  
│ │  
│ ├─ Settings/  
│ │ ├─ ScriptableObjects/  
│ │ └─ Input/  
│ │  
│ ├─ Art/  
│ ├─ Audio/  
│ ├─ Editor/  
│ └─ AddressableAssetsData/  
│  
└─ ThirdParty/

## **3. 최상위 폴더 규칙 (Top-Level Directory Rules)**

| **폴더명** | **설명 및 규칙** |
| --- | --- |
| **\_Project/** | **프로젝트의 모든 고유 에셋을 포함하는 루트 폴더입니다.** \_ 접두사는 Unity 에디터 상에서 항상 최상단에 위치하도록 하여 접근성을 높입니다. 서드파티 에셋은 이 폴더에 절대 위치해서는 안 됩니다. |
| **ThirdParty/** | **Asset Store 등 외부에서 가져온 모든 패키지와 에셋을 관리하는 폴더입니다.** 각 패키지는 벤더(Vendor)나 패키지 이름으로 된 하위 폴더에 저장됩니다. (예: ThirdParty/DOTween/) |

## **4. \_Project/ 하위 폴더 상세 규칙**

### **4.1. Features/ - 핵심 기능 모듈**

이 폴더는 게임의 모든 기능적 모듈을 포함하며, 이 프로젝트 구조의 심장입니다.

* **원칙 1 (기능 단위):** 각 하위 폴더는 독립적인 기능 단위(Feature)입니다. 특정 기능과 관련된 모든 종류의 에셋(스크립트, 프리팹, 데이터 등)은 해당 기능 폴더 내에 위치하는 것을 원칙으로 합니다.
* **원칙 2 (Art/Audio):** **모든 기능 폴더는 필요 시 내부에 Art/ 또는 Audio/ 폴더를 생성하여 기능 전용 리소스를 관리할 수 있습니다.** 이는 기능의 응집도를 높이기 위함이며, 여러 기능에서 공유되는 리소스는 루트의 범용 Art/, Audio/ 폴더를 사용합니다.
* **원칙 3 (Prefabs):** 기능 폴더 내 Prefabs/ 폴더의 규모가 커질 경우, **[유형/역할] 패턴에 따라 하위 폴더를 생성**하여 관리하는 것을 적극 권장합니다.

| **기능 폴더** | **책임 (Responsibility)** | **주요 내용물 및 프리팹 권장 구조** |
| --- | --- | --- |
| **Core/** | 게임 전체에 영향을 미치는 필수 공통 시스템. | GameManager, SceneLoader, **Prefabs/Managers/** |
| **Dungeon/** | 던전 방의 생성, 배치, 상호작용 등 인게임 필드 관련 기능. | DungeonManager, RoomController, **Prefabs/Rooms/, Prefabs/Traps/** |
| **Combat/** | **전투 '행위'에 직접 관련된 요소만 관리합니다.** 데미지 계산, 투사체, VFX 등. | DamageCalculator, **Prefabs/Projectiles/, Prefabs/VFX/** |
| **Units/** | **모든 유닛(몬스터, 마왕)의 근간이 되는 데이터와 로직.** | InGameUnit 클래스, **Prefabs/Characters/, Prefabs/Summons/** |
| **InventorySystem/** | 인게임에서 획득하는 아이템(룬, 스킬북 등)의 관리 기능. | InventoryManager, **Prefabs/Slots/, Prefabs/DroppedItems/** |
| **Equipment/** | 영구/인스턴스 장비의 데이터, 로직, 강화 시스템. | EquipmentManager, **Prefabs/Icons/, Prefabs/Effects/** |
| **MetaGame/** | 게임의 주된 루프 바깥에서 이루어지는 모든 기능. (로비, 영구 성장) | 강화/가챠 시스템, **Prefabs/Lobby/, Prefabs/GachaEffects/** |
| **UI/** | **중앙화된 UI 관리 시스템.** 모든 UI 에셋은 이곳에서 관리하여 일관성을 유지합니다. | UIManager, ViewModels, UXML/USS 파일, 커스텀 컨트롤 |

* **UI 폴더 상세 규칙:** Features/UI/ 폴더 내부는 다른 기능 폴더 구조를 반영하여 하위 폴더를 생성합니다.
  + **Common/ 폴더의 역할:** UI/UXML/Common/ 과 UI/Scripts/ViewModels/Common/ 은 여러 기능에 걸쳐 재사용되는 **UI 문서 및 로직**(예: 공용 팝업, 로딩 화면, 설정 창)을 관리합니다.

Features/UI/  
├─ Scripts/  
│ └─ ViewModels/  
│ ├─ Dungeon/  
│ ├─ MetaGame/  
│ ├─ Equipment/  
│ ├─ InventorySystem/  
│ └─ Common/  
├─ UXML/  
│ ├─ Dungeon/  
│ ├─ MetaGame/  
│ ├─ Equipment/  
│ ├─ InventorySystem/  
│ └─ Common/  
└─ USS/  
 ├─ Themes/  
 └─ Components/

### **4.2. Settings/ - 전역 설정 데이터**

여러 기능에 걸쳐 공유되거나, 게임 전체의 동작을 결정하는 설정 에셋을 관리합니다.

| **폴더명** | **책임 (Responsibility)** | **주요 내용물** |
| --- | --- | --- |
| **ScriptableObjects/** | **전역적으로 참조되는 ScriptableObject 데이터.** | BalanceData/ (밸런스 공식), EventChannels/ (시스템 간 통신 이벤트) |
| **Input/** | Unity의 Input System을 위한 입력 액션 에셋. | PlayerInputActions.inputactions |

### **4.3. Art/ & Audio/ - 범용 에셋**

여러 기능에서 공통으로 사용될 수 있는 범용 아트/오디오 에셋을 관리합니다.

* **규칙:** 특정 기능에만 종속되는 에셋은 해당 기능 폴더 내에 위치시켜야 합니다.
* **Common의 역할:** 이 폴더들 내의 Common 접미사는 특정 게임 기능에 종속되지 않고, **UI와 게임 월드 전역에서 사용될 수 있는 시각/청각 리소스**를 의미합니다.
* **권장 하위 구조:**  
  Art/  
  ├─ Materials/  
  ├─ Shaders/  
  └─ SpritesCommon/ # 공용 아이콘, UI 배경 등  
    
  Audio/  
  ├─ BGM/  
  ├─ SFXCommon/ # 여러 곳에서 쓰이는 효과음  
  └─ UI/ # UI 전용 효과음  
   ├─ Click/  
   ├─ Popup/  
   └─ Notification/

### **4.4. 기타 폴더**

| **폴더명** | **책임 (Responsibility)** |
| --- | --- |
| **Editor/** | 모든 커스텀 에디터 확장 스크립트. (예: 커스텀 인스펙터, 데이터 임포터) |
| **AddressableAssetsData/** | Unity Addressable Asset System이 생성하고 관리하는 설정 파일. **Git에 커밋해야 하는 파일은 AddressableAssetSettings.asset 및 \*.bin 설정 파일들이며, 빌드 시 생성되는 \*.hash, \*.json 파일 등은 .gitignore 처리하거나 Git LFS로 관리합니다. (주의: 이 폴더의 하위 구조는 Unity에 의해 자동으로 관리되므로, 직접 수정하지 마십시오.)** |

## **5. 네이밍 컨벤션 (Naming Convention)**

일관된 네이밍 컨벤션은 검색 효율성을 높이고 에셋의 역할을 명확하게 합니다.

### **5.1. 일반 규칙**

* **파스칼 케이스 (PascalCase):** 모든 파일과 폴더 이름은 파스칼 케이스를 사용합니다. (예: UnitData, SpritesCommon)
* **공백 금지:** 이름에 공백을 사용하지 않습니다.
* **명확성:** 약어보다는 명확하고 서술적인 이름을 사용합니다. (예: CharCtrl (X) → CharacterController (O))
* **CI/CD 환경 주의사항:** **모든 파일과 폴더명은 대소문자를 엄격하게 구분하여 작성합니다.** Windows와 Mac/Linux 간의 파일 시스템 차이로 인해 발생하는 경로 문제를 예방하기 위함이며, 이는 CI/CD 파이프라인의 안정성에 필수적입니다.

### **5.2. 에셋 타입별 접두사/접미사**

| **에셋 타입** | **규칙** | **예시** |
| --- | --- | --- |
| **ScriptableObject** | **SO\_ 접두사를 사용하고, 역할과 이름을 \_로 구분합니다.** | SO\_UnitData\_Goblin<br>SO\_EquipmentData\_Permanent\_Excalibur<br>SO\_Event\_OnEnemyKilled<br>SO\_Balance\_ThreatFormula |
| **Scene** | 접미사 없이, 명확한 역할 이름만 사용합니다. | Initialization, Lobby, MainGame |
| **Prefab** | **PF\_ 접두사를 사용하여 검색 편의성을 높입니다.** | PF\_Goblin, PF\_FireballVFX, PF\_ConfirmPopup |
| **Material** | M\_ 접두사를 사용합니다. | M\_Character, M\_Environment |
| **Texture** | T\_ 접두사와 용도에 따른 접미사를 사용합니다. | T\_Goblin\_D (Diffuse/Albedo)<br>T\_UI\_Icon\_Heart |
| **Animation Clip** | **소속 기능, 대상, 동작을 \_로 구분합니다.** | Units\_Goblin\_Attack<br>MetaGame\_Enhancement\_Success |
| **Animator Controller** | AC\_ 접두사를 사용합니다. | AC\_Goblin |

## **6. 최종 폴더 구조 요약 (Final Structure Summary)**

아래는 본 문서에서 정의한 모든 규칙을 반영한 전체 폴더 구조의 시각적 요약입니다.

Assets  
└── \_Project  
 ├── AddressableAssetsData  
 ├── Art  
 │ ├── Materials  
 │ ├── Shaders  
 │ └── SpritesCommon  
 ├── Audio  
 │ ├── BGM  
 │ ├── SFXCommon  
 │ └── UI  
 │ ├── Click  
 │ ├── Notification  
 │ └── Popup  
 ├── Editor  
 ├── Features  
 │ ├── Combat  
 │ │ ├── Audio  
 │ │ ├── Prefabs  
 │ │ │ ├── Projectiles  
 │ │ │ └── VFX  
 │ │ └── Scripts  
 │ ├── Core  
 │ │ ├── Prefabs  
 │ │ │ └── Managers  
 │ │ ├── Scenes  
 │ │ └── Scripts  
 │ ├── Dungeon  
 │ │ ├── Art  
 │ │ ├── Prefabs  
 │ │ │ ├── Rooms  
 │ │ │ └── Traps  
 │ │ ├── ScriptableObjects  
 │ │ └── Scripts  
 │ ├── Equipment  
 │ │ ├── Art  
 │ │ ├── Prefabs  
 │ │ │ ├── Effects  
 │ │ │ └── Icons  
 │ │ ├── ScriptableObjects  
 │ │ └── Scripts  
 │ ├── InventorySystem  
 │ │ ├── Audio  
 │ │ ├── Prefabs  
 │ │ │ ├── DroppedItems  
 │ │ │ └── Slots  
 │ │ ├── ScriptableObjects  
 │ │ └── Scripts  
 │ ├── MetaGame  
 │ │ ├── Audio  
 │ │ ├── Prefabs  
 │ │ │ ├── GachaEffects  
 │ │ │ └── Lobby  
 │ │ ├── Scenes  
 │ │ ├── ScriptableObjects  
 │ │ └── Scripts  
 │ ├── UI  
 │ │ ├── Scripts  
 │ │ │ └── ViewModels  
 │ │ │ ├── Common  
 │ │ │ ├── Dungeon  
 │ │ │ ├── Equipment  
 │ │ │ ├── InventorySystem  
 │ │ │ └── MetaGame  
 │ │ ├── USS  
 │ │ │ ├── Components  
 │ │ │ └── Themes  
 │ │ └── UXML  
 │ │ ├── Common  
 │ │ ├── Dungeon  
 │ │ ├── Equipment  
 │ │ ├── InventorySystem  
 │ │ └── MetaGame  
 │ └── Units  
 │ ├── Animations  
 │ ├── Audio  
 │ ├── Prefabs  
 │ │ ├── Characters  
 │ │ └── Summons  
 │ ├── ScriptableObjects  
 │ └── Scripts  
 └── Settings  
 ├── Input  
 └── ScriptableObjects  
 ├── BalanceData  
 └── EventChannels