

팩맨

네오플 제주 아카데미
김지헌 교육생



UI 요소



< 인트로 화면 >



< 로딩 화면 >

UI 요소

[메인 화면]

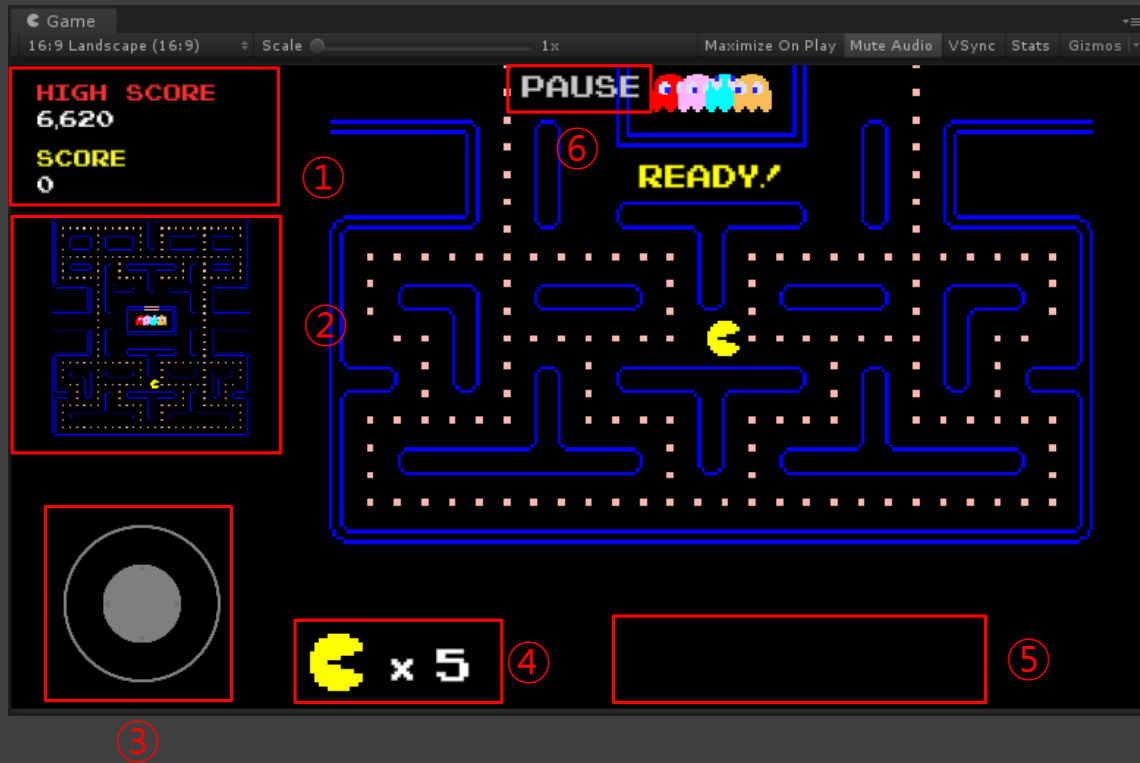
1. 최고 점수
2. 스테이지 선택
3. 스테이지 플레이 & 종료

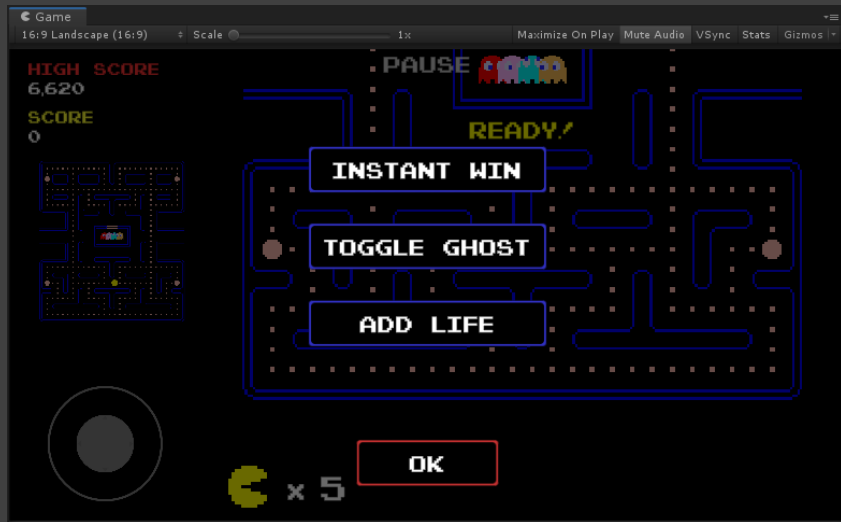


UI 요소

[게임 플레이]

1. 최고 점수 & 현재 점수
2. 미니 맵
3. 가상 조이스틱
4. 남은 목숨
5. 지금까지 먹은 과일
6. 일시정지 메뉴





UI 요소

[일시 정지 메뉴]

1. 계속
2. 치트 메뉴
3. 나가기

[치트 메뉴]

1. 즉시 클리어
2. 유령 활성화/비활성화
3. 목숨 추가

```
public class MenuUIManager : Singleton<MenuUIManager>
{
    public Action PreStageIndexAction;
    public Action ProStageIndexAction;
    public Action StartStageAction;
}
```

참조 1개

```
void AddMenuFunction()
{
    MenuUIManager.Instance.ProStageIndexAction += IncreaseStageIndex;
    MenuUIManager.Instance.PreStageIndexAction += DecreaseStageIndex;
    MenuUIManager.Instance.StartStageAction += StartNewGame;
}
```

참조 0개

```
public void ButtonEvent_PreStageIndex()
{
    PreStageIndexAction?.Invoke();
    AudioManager.Instance.PlaySound(ESfxId.Click);
}
```

참조 0개

```
public void ButtonEvent_ProStageIndex()
{
    ProStageIndexAction?.Invoke();
    AudioManager.Instance.PlaySound(ESfxId.Click);
}
```

참조 0개

```
public void ButtonEvent_StartStage()
{
    AudioManager.Instance.PlaySound(ESfxId.Click);
    StartStageAction?.Invoke();
    LoadingSceneManager.LoadScene(SceneName.StageSceneName);
}
```

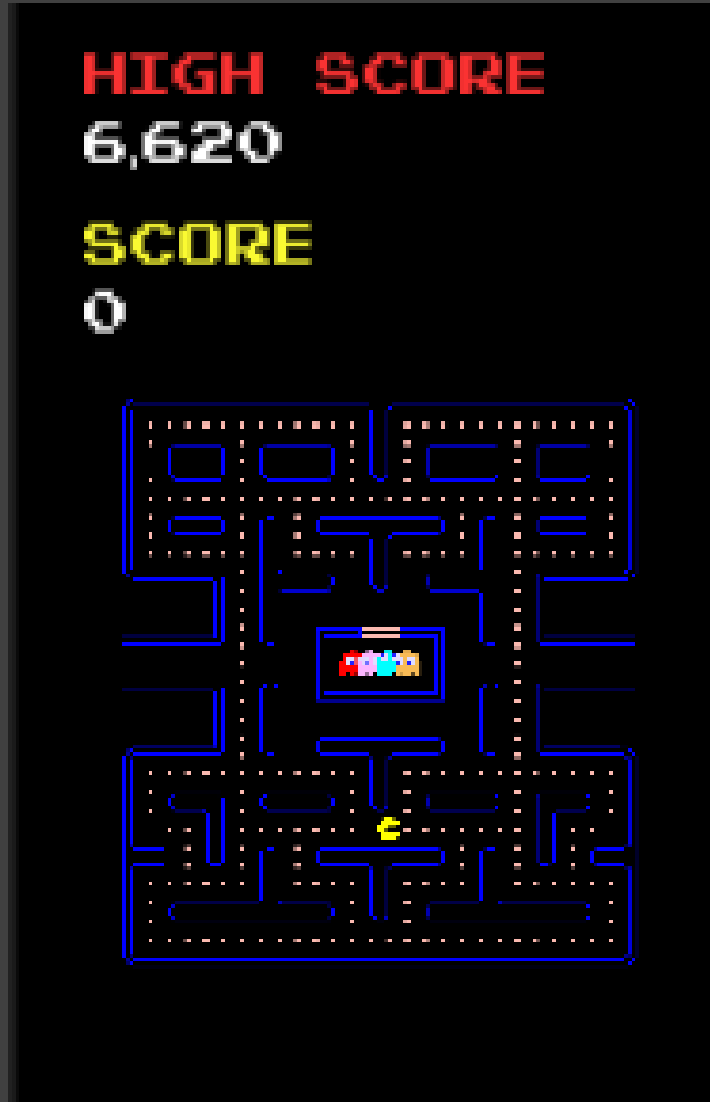
UI 요소

- UI 상호 작용은 System.Action으로 구현

1. UI 매니저에 Action 선언

2. 다른 클래스에서 UI 매니저 Action에 함수 체인

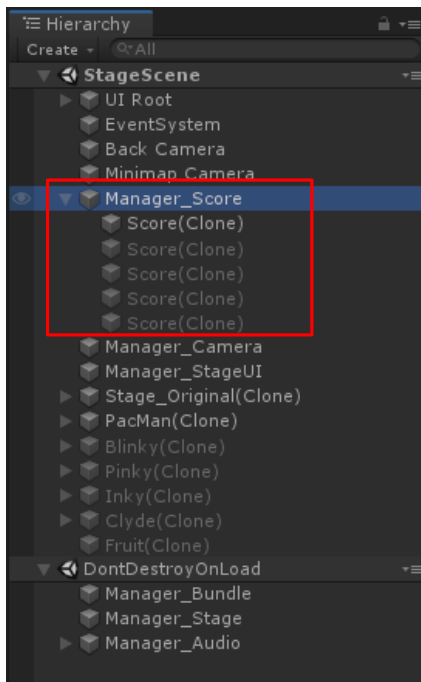
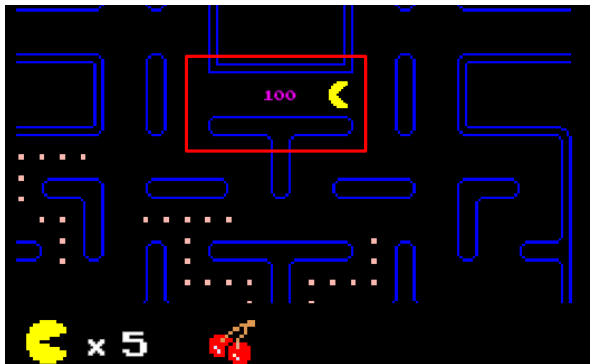
3. UI 상호작용 시에 Null 검사 후 함수 체인 실행



UI 요소

[미니맵]

- 렌더 텍스처로 구현
- 미니 맵 카메라가 전체 맵을 렌더링



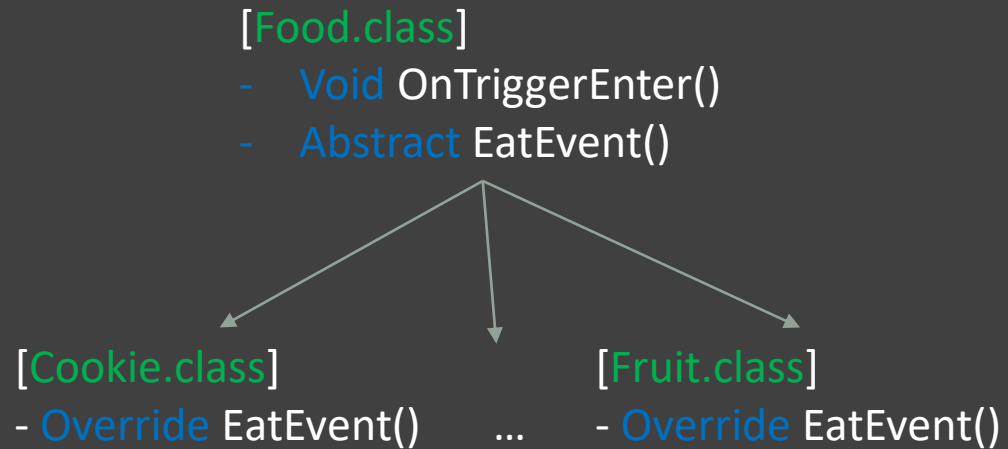
UI 요소

[스코어 정보]

- 유령과 과일을 먹을 때 스코어
오브젝트를 띄워 줌

- 스코어 오브젝트는
오브젝트 풀을 이용해 관리

음식 오브젝트



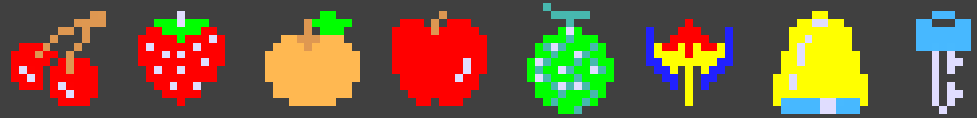
[추상 Food Class]

- 공용 부분인 OnTriggerEnter() 구현
- OnTriggerEnter()에서 EatEvent() 함수 호출

[자식 Food Class]

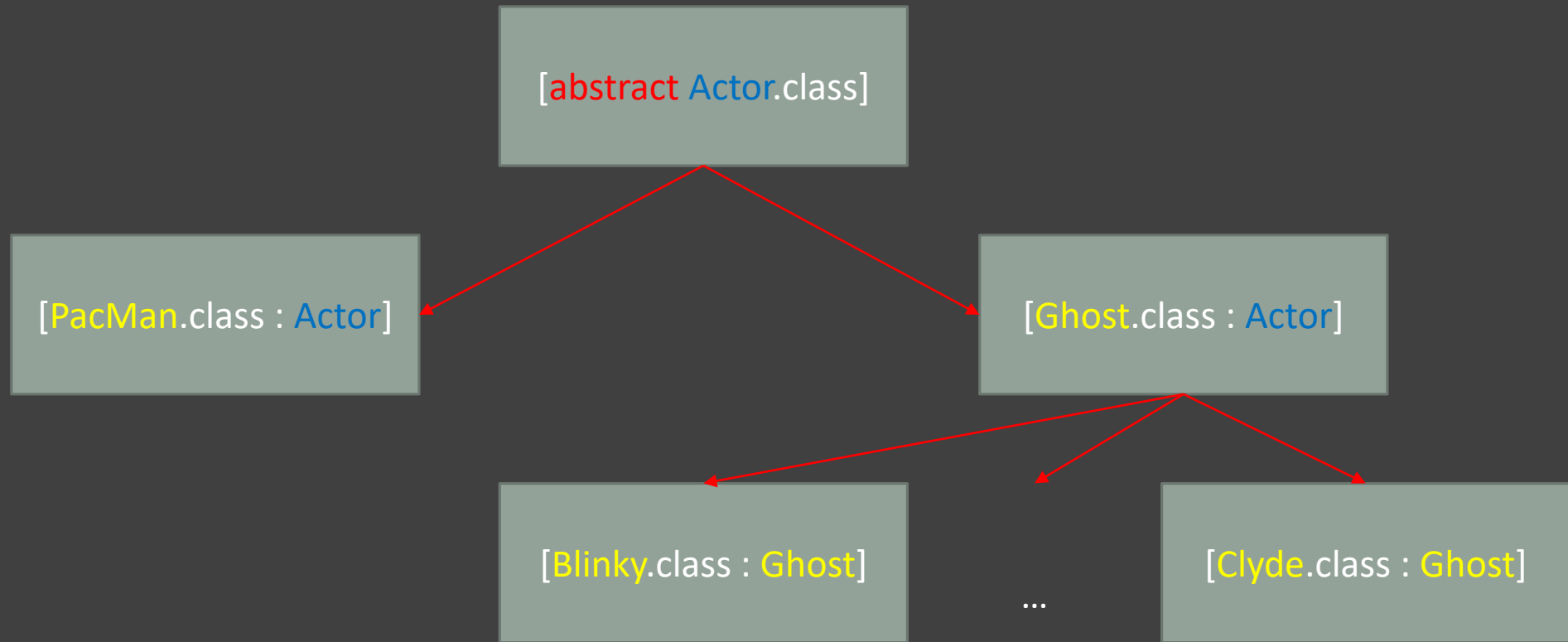
- 추상 EatEvent()를 override해 기능 구현

과일 오브젝트

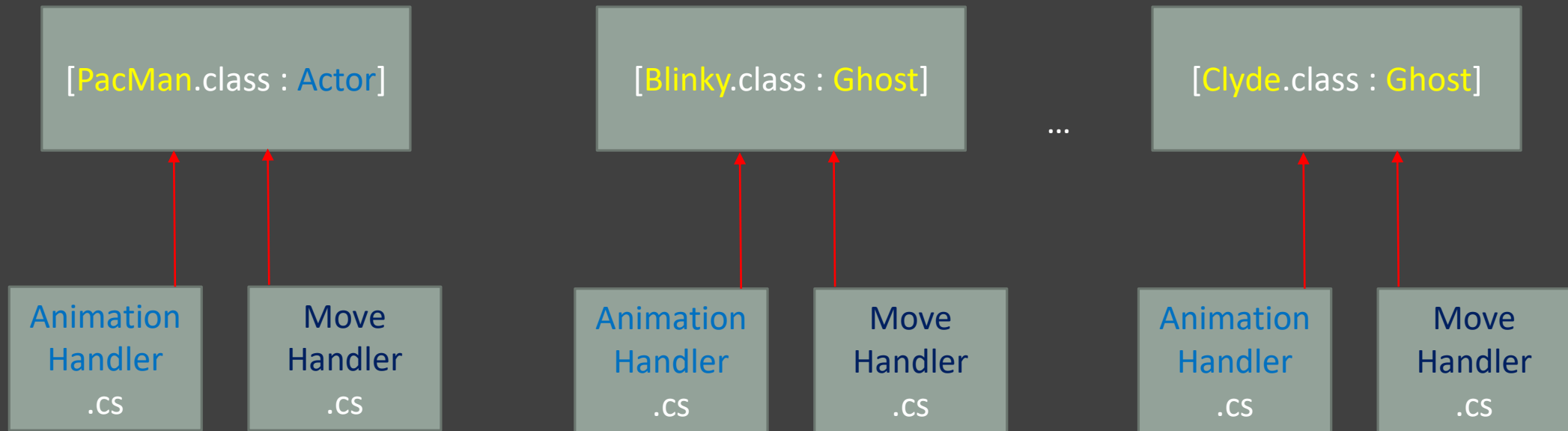


- 과일은 스테이지가 진행 될 수록 높은 점수를 줌
- 최소 100 에서 최대 5,000
- 스프라이트는 Atlas로 관리

Actor 클래스 상속



Handler 재사용



이동 선 입력

- PacMan(플레이어)는 '현재 이동 방향'과 '예약 이동 방향' 2가지 방향을 가지고 있음
 - 플레이어가 방향키를 입력하면 '예약 이동 방향'을 업데이트
1. 타일 이동이 끝나면 '예약 이동 방향'으로 이동할 수 있는지 확인
 2. '예약 이동 방향'으로 이동할 수 없으면 '현재 이동 방향'으로 계속 이동
 3. '현재 이동 방향'으로도 이동할 수 없으면 이동 종료

A* 알고리즘

- 유령의 목적지 도착 경로는 A* 알고리즘으로 계산
- A* 알고리즘은 `Update()` 마다 호출하지 않음
- A* 알고리즘은 각 유령 객체 마다 호출 시점이 다름
 1. 새로운 목적지를 부여 받을 때
 2. 팩맨을 추적하는 상태 & 타일 이동을 완전히 끝냈을 때

유령 행동 규칙



- [블링키]
- 팩맨의 **뒤**를 쫓는다
 - 주눅 모드(1시 방향 도주)



- [잉키]
- 팩맨과 **점대칭 위치**로 이동
 - 주눅 모드(5시 방향 도주)

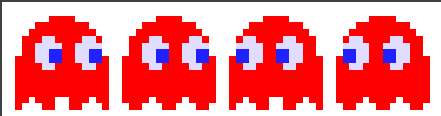


- [핑키]
- 팩맨의 **앞지르기** 시도
 - 주눅 모드(11시 방향 도주)



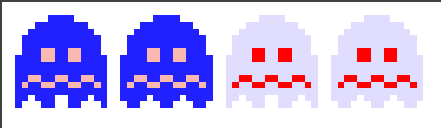
- [클라이드]
- 아무 생각이 없다(**랜덤**)
 - 주눅 모드(7시 방향 도주)

유령 행동 상태



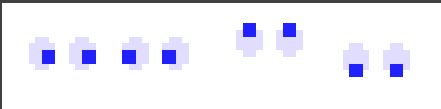
1. 노멀(Normal)

- 평범한 상태



2. 추적(Tracking)

- 플레이어를 잡기 위해 바로 쫓음
- 워프 게이트 사용



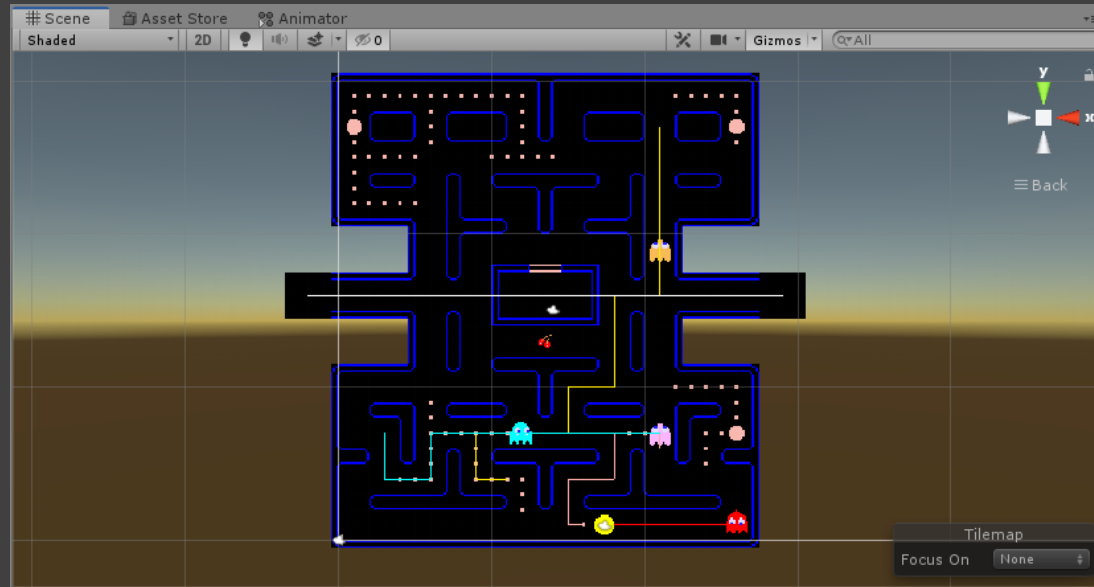
3. 주눅(Timid)

- 플레이어를 피해 흩어짐

4. 복귀(Retreat)

- 주눅 상태에서 팩맨에게 당하면 집으로 복귀

이동 경로 디버그



< Gizmo를 사용해 유령의 이동 방향 확인 가능 >

스테이지 리스트

```
[
  {
    "stageIndex": 0,
    "stageName": "Stage 0",
    "stagePath": "Assets/Prefab/Stage/Stage_Original.prefab"
  },
  {
    "stageIndex": 1,
    "stageName": "Stage 1",
    "stagePath": "Assets/Prefab/Stage/Stage_ChampionDX.prefab"
  },
  {
    "stageIndex": 2,
    "stageName": "Stage 2",
    "stagePath": "Assets/Prefab/Stage/Stage_ChampionDX_2.prefab"
  }
]
```

- ✓ JSON으로 스테이지 데이터를 읽어 들인다
- ✓ StageManager에서 List로 관리

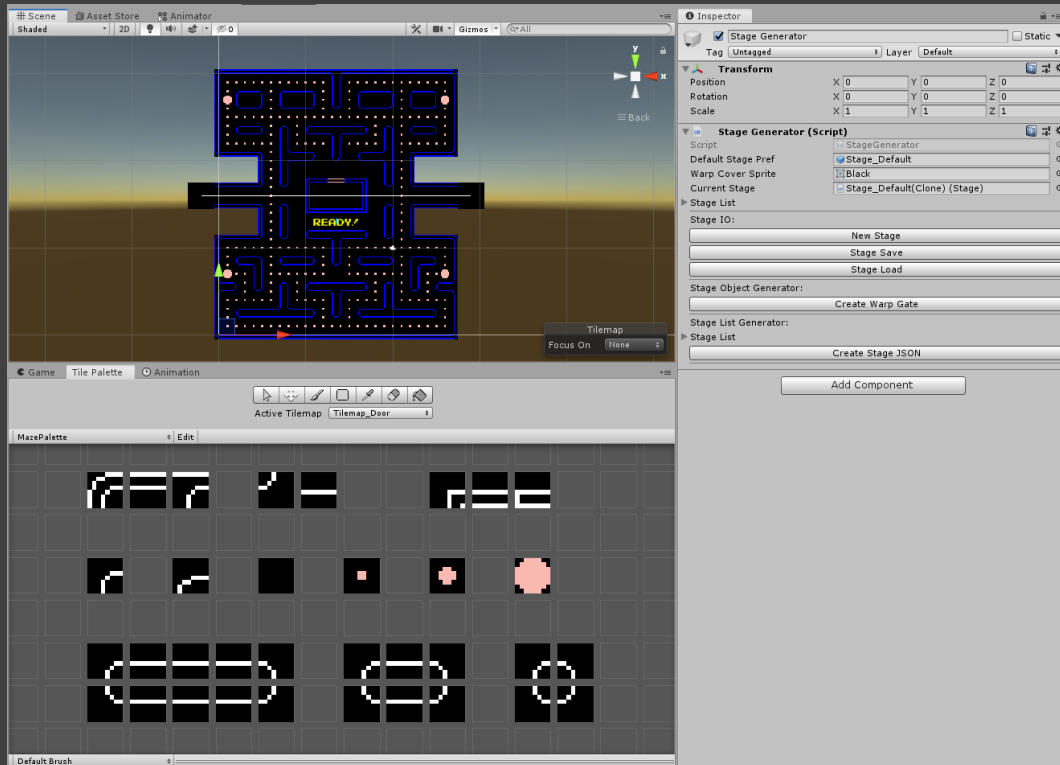
스테이지 매니저

- Singleton 클래스
- 현재 게임 상태 관리
- 현재 스테이지 정보를 가지고 있음
- 연속된 스테이지 진행을 위해 씬이 로딩되어도 **삭제하지 않음**
- 스테이지 정보에서 팩맨과 유령의 시작 위치를 읽어와 Instantiate
- 먹은 과일 리스트 관리

```
참조 19개
public enum EState
{
    Reset,
    Prepare,
    Play,
    PacManDie,
    GameOver,
    StageOver
}
```

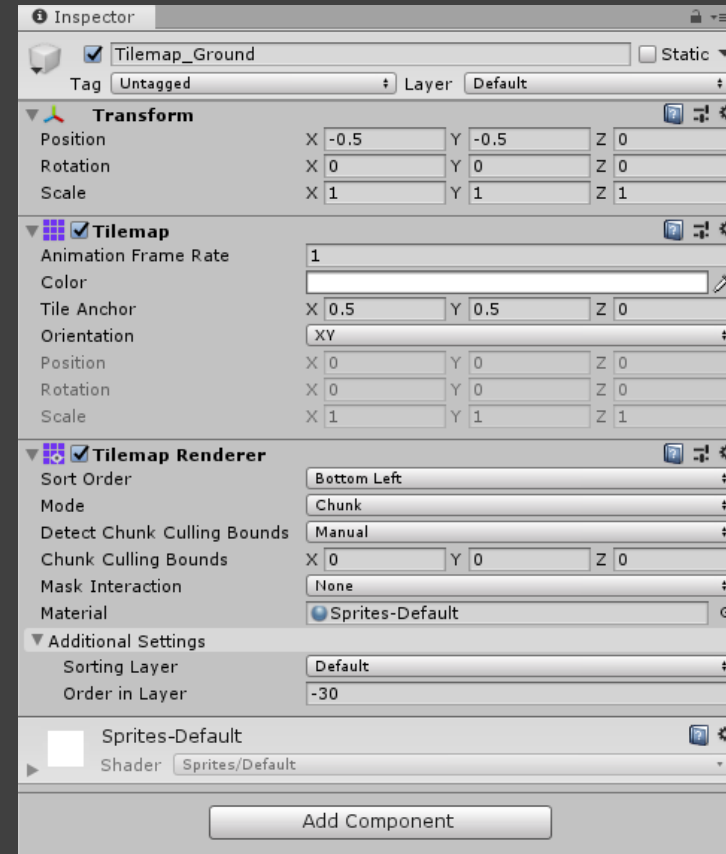
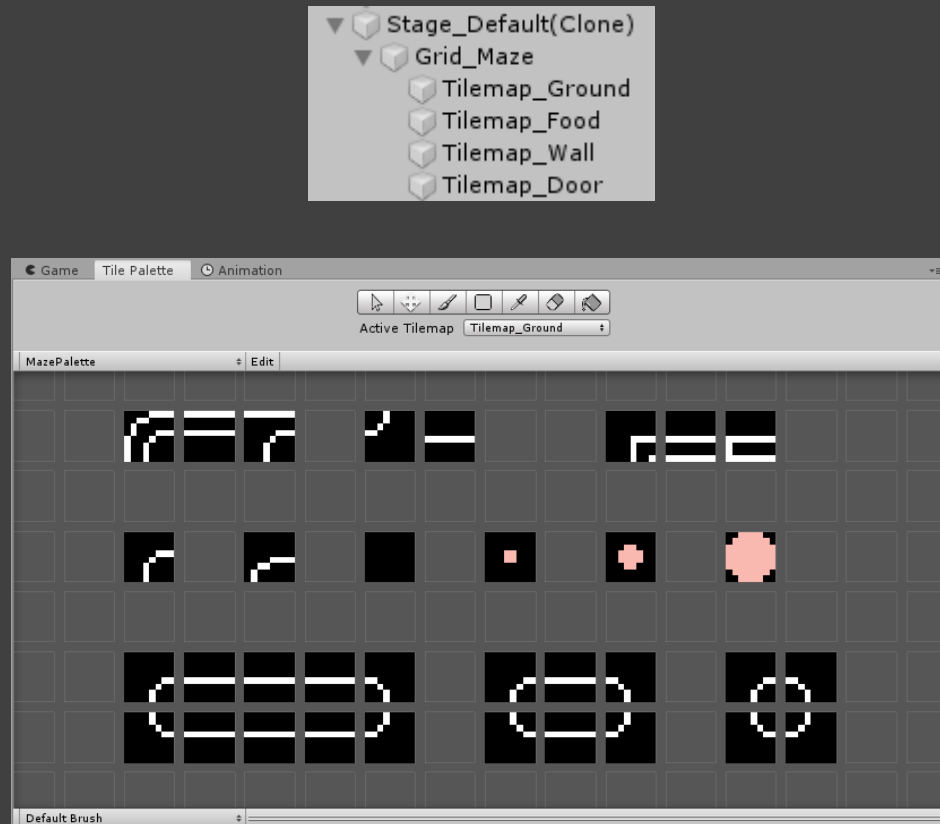
<게임 상태 Enum>

맵 에디터

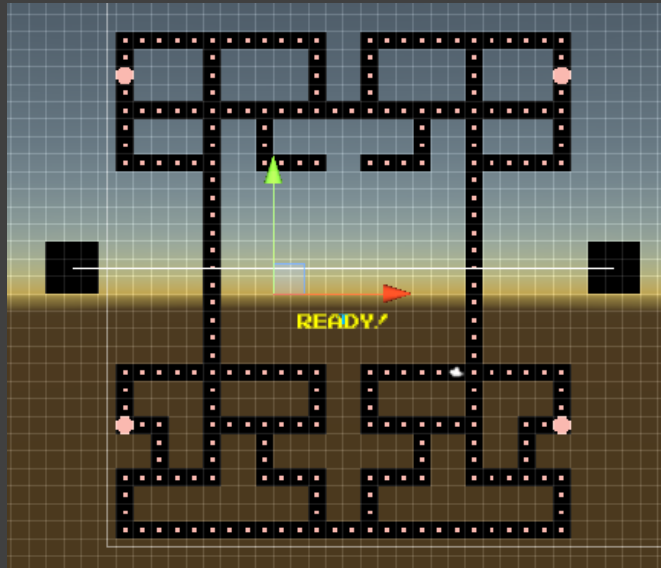


- 별도의 씬에서 에디터로 맵 편집
- 새로 만들기, 불러오기, 저장 가능
- 워프 게이트 등 게임 오브젝트 생성
- 만들어진 맵 리스트를 JSON으로 추출

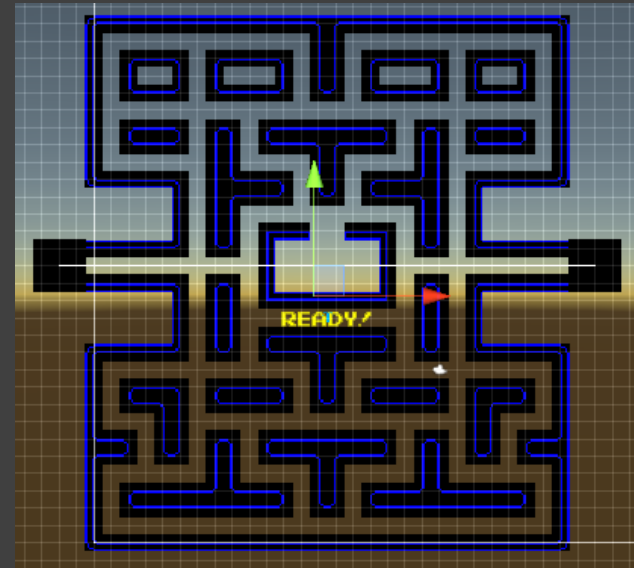
타일 맵



타일 맵

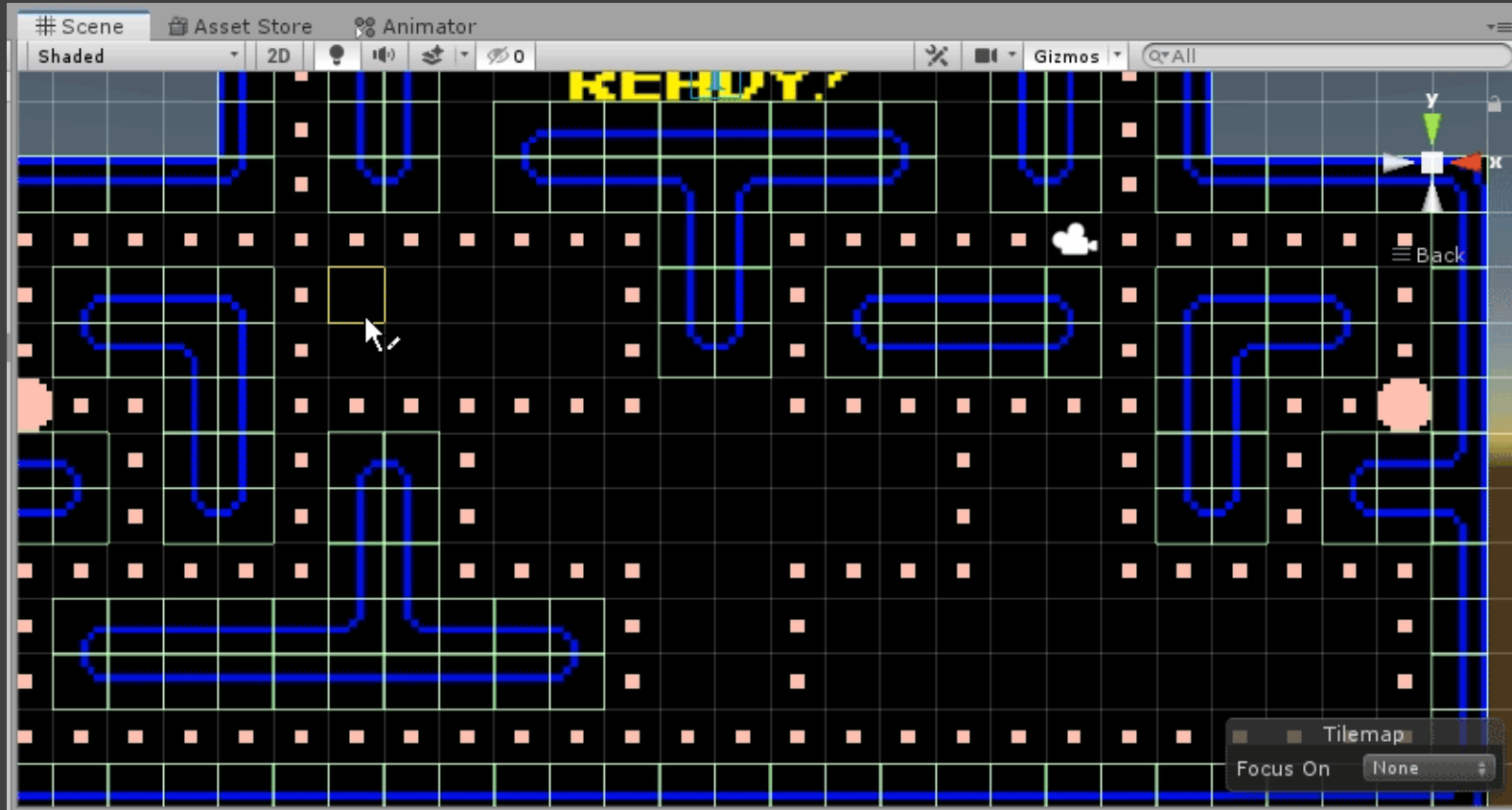


< 음식 타일만 Active >

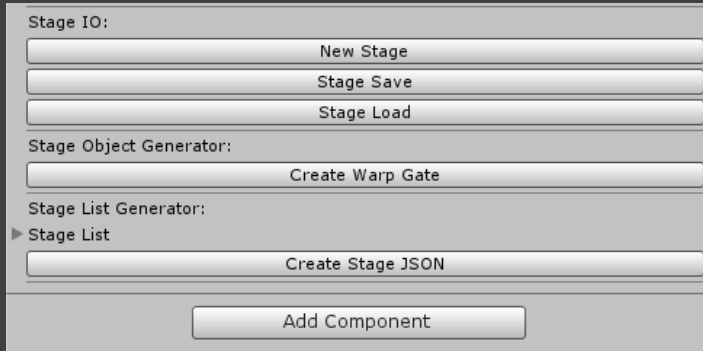


< 벽 타일만 Active >

Rule(규칙) 타일



커스텀 인스펙터



[StageGenerator.cs]

- MonoBehaviour 상속

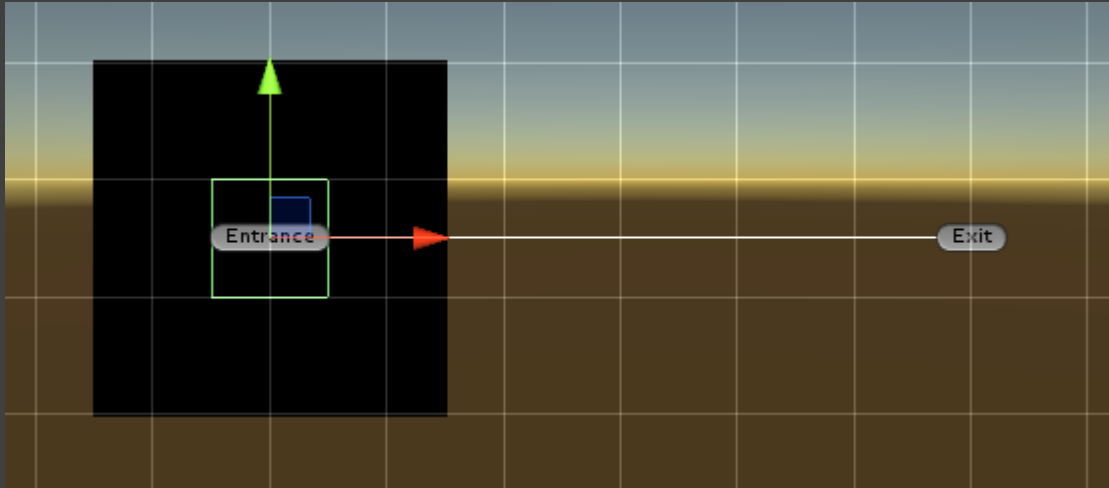


[CustomEditor(typeof(StageGenerator))]

[StageIOButton.cs]

- Editor 상속
- OnInspectorGUI() 오버라이드

워프 게이트



- 커스텀 인스펙터로 워프 게이트 생성
- 생성된 워프게이트는 Gizmo로
시각적 편집 가능
- Grid와 연동되어 있어 칸 단위로 이동

에셋 번들

The screenshot displays the Firebase Storage interface for a project named 'NeoLife'. The left sidebar contains navigation links for various Firebase services: Authentication, Database, Storage (selected), Hosting, Functions, and ML Kit. Below these are sections for '품질' (Quality), '애널리틱스' (Analytics), and '성장' (Growth). The main content area shows the 'Storage' tab with a list of files under the path 'gs://neolife-c0ce4.appspot.com > AssetBundle'. The table lists seven files with their names, sizes, MIME types, and last update dates. A '파일 업로드' (Upload File) button is visible in the top right of the file list.

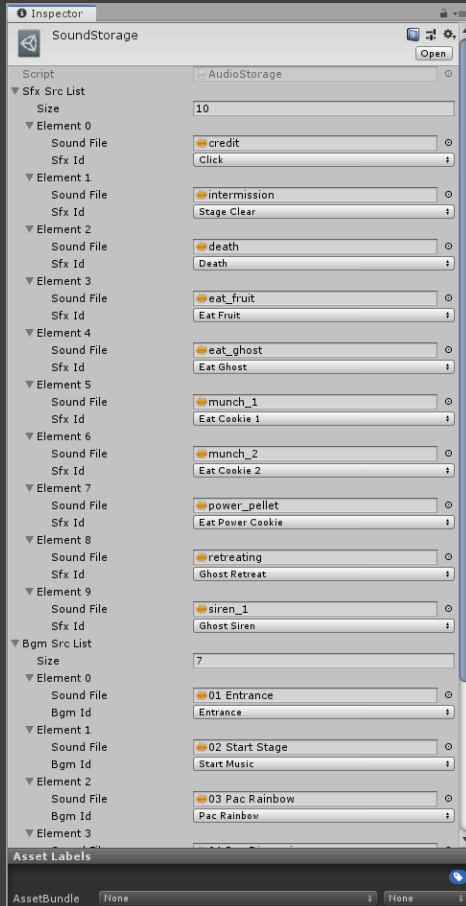
<input type="checkbox"/>	이름	크기	유형	최종 수정 날짜
<input type="checkbox"/>	characterandequipment	178.96 KB	application/octet-stream	2019. 12. 29.
<input type="checkbox"/>	characterandequipment_win	144.03 KB	application/octet-stream	2020. 1. 2.
<input type="checkbox"/>	farmasset	50.74 KB	application/octet-stream	2019. 12. 29.
<input type="checkbox"/>	farmasset_win	69.63 KB	application/octet-stream	2020. 1. 2.
<input type="checkbox"/>	stage_asset_android	70.98 KB	application/octet-stream	2020. 1. 13.
<input type="checkbox"/>	stage_asset_pc	62.79 KB	application/octet-stream	2020. 1. 13.

에셋 번들



1. 인트로 씬에서 비동기 로딩 씬을 로드
2. 구글 파이어 베이스에서 에셋 번들 다운로드
3. 다운로드된 에셋 번들 캐싱
4. 다운로드 및 캐싱이 완료되면 메뉴 씬 로드

사운드



- 사운드 파일은 스크립터블 오브젝트로 관리
- 스크립터블 오브젝트는 별도의 Sound Manager에 캐싱

```
public enum ESfxId
{
    Click,
    StageClear,
    Death,
    EatFruit,
    EatGhost,
    EatCookie1,
    EatCookie2,
    EatPowerCookie,
    GhostRetreat,
    GhostSiren,
}
```

< 효과음 enum >

```
public enum EBgmId
{
    Entrance,
    StartMusic,
    PacRainbow,
    PacDimensions,
    PacAvenue,
    PacLogic,
    PacManCeBgm,
    End
}
```

< 배경음 enum >

- 끝 -