프로젝트 중간보고서

(프로젝트 명 : 루카스 어드벤처)

32153033 이건희, 32154024 장현준

[목차]

1. 프로젝트 배경

1.1 제안 배경

- 게임 개발에 대한 흥미와 개발 과정에 대한 전체적인 이해하기 위함
- 새로운 장르의 개발에 대한 도전

1.2 기능 개요

- 로그라이크 RPG
- 주어진 스테이지를 통과하면서 점점 강력해지는 캐릭터를 조작하며 끝내 마지막 보스를 처치하는 진행 방식
- 탑다운 뷰를 통한 종-횡 이동이 주를 이루는 조작 방식
- 기본적인 인터페이스(플레이어 상태바, 소지품창)를 지원하며 더 직관적인 상태를 파악할 수 있다.
- 구현된 상점에서 몬스터를 처치하고 나온 재화 또는 아이템을 활용하여 더욱 높은 단계의 아이템을 착용함으로 플레이어의 성취감을 만들어 낸다.
- 맵 구성, 아이템의 드랍 유무, 아이템의 희소성에 따른 무작위 드랍 으로 인해 플레이어는 매번 게임을 즐길때 마다 다양한 조합을 느낄 수 있다.

1.3 기대 효과

- 2D 게임, Unity, C# 인터페이스 이해도 증가
- Github활용으로 인한 이해도 증가와 협업 능력의 향상

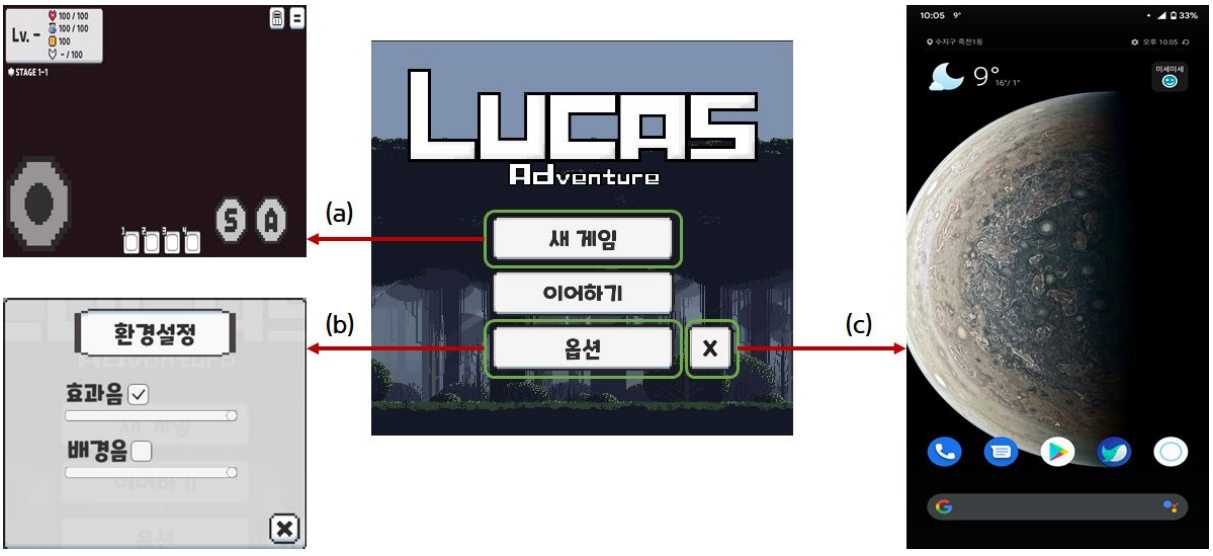
2. 사용자 가이드

2.1 설치 및 사용 방법

1) **설치 :** APK파일을 다운로드 후 설치하여 게임을 실행한다.
2) **사용방법**
    a) 시작하는 장소(home)에서 클리어하고 싶은 영역(Area) 입구로 향한다.
    b) 각 Area속 스테이지에 있는 몬스터들을 공격키를 사용하여 사냥하여 레벨을 올리고 아이템도 줍는다.
    c) 보스 스테이지를 클리어하면 home으로 다시 시작한다.
    d) 스테이지나 보스를 클리어하지 못하면 다시 home으로 시작한다.
    e) 다른 스테이지를 도전하기 전에 필요한 아이템을 상점에 구매하여 재정비를 한다.
    f) 모든 영역의 전부 클리어하면 게임이 끝난다.

요구사항 명세

1)게임 시작 화면



(a) : 새로 게임을 시작하는 버튼으로, 클릭 시 로딩화면을 지나 게임 내부의 시작지점으로 연결된다.
(b) : 환경설정 창을 활성화하는 버튼으로, 효과음과 배경음을 조절할 수 있는 창이 나온다.
(c) : 나가기 버튼으로, 클릭 시 어플이 종료되어 모바일 기기의 홈화면으로 나가게 된다.

2) 아이템

무기		
근접	검	근접한 거리에서 공격하는 무기
	도끼	일정반경의 적을 전부 공격하는 무기
	창	검보다 긴 사거리로 공격 가능한 무기
기타 아이템		
물약		HP 또는 MP를 회복시켜주는 물약
Coin		상점에 이용이 가능한 재화
고유 아이템		플레이어에 능력을 부여하거나 강화시키는 아이템

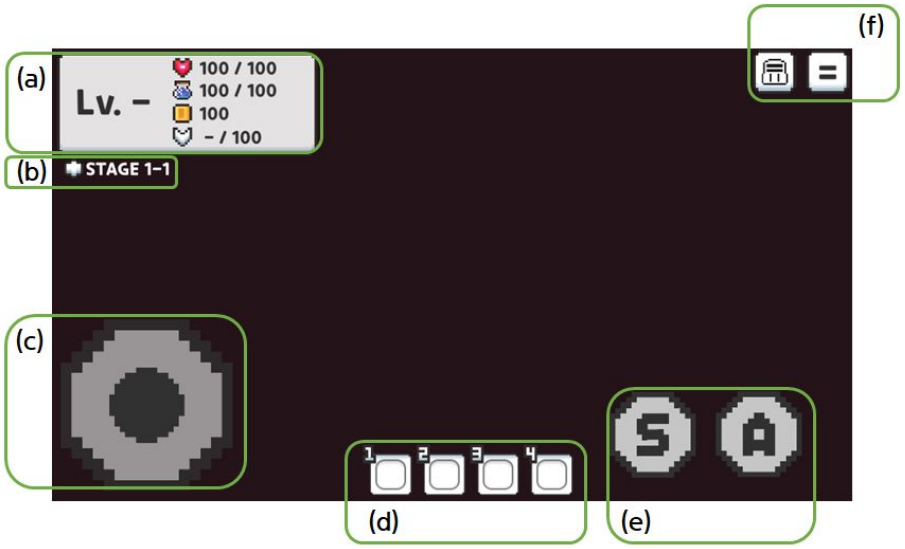
3) 상점

(a) : 아이템 탭으로 구매하고자하는 아이템 종류를 선택합니다

(b) : 구매가능한 아이템을 나열하는 뷰입니다. 구매하고자하는 아이템을 클릭하여 선택합니다.

(c) : 의사에 따라 구매를 결정 또는 취소할 수 있습니다.

4)게임 UI 구성



[인게임 UI]

(a) : 플레이어의 상태를 나타내는 그룹으로 HP, MP, EXP, 코인, 레벨을 확인할 수 있다.

(b) : 현재 들어와있는 맵의 정보를 나타낸다.

(c) : 플레이어의 이동을 담당하는 가상패드이다.

(d) : 플레이어의 무기 슬롯이다. 사용자가 습득한 무기가 순서대로 등록된다. 사용이 완료된 무기는 사라지고 후방에 위치한 무기들은 앞으로 당겨진다.

(e) : 플레이어의 액션을 담당하는 버튼이다. 무기의 교체를 실행하는 S버튼과 공격을 실행하는 A버튼이 있다.

(f) : 화면에 다른 창을 띄우는 버튼 그룹이다. 인벤토리 창과 메뉴 창을 띄우는 기능을 한다.

[인게임 메뉴 화면]

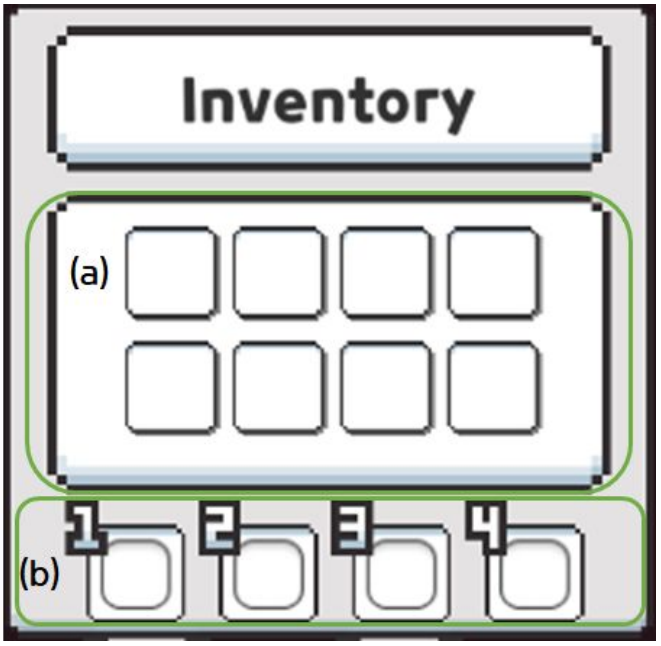
(a) : 현재 진행상황을 저장하는 버튼이다. 현재의 스테이지 클리어 현황, 플레이어의 상태, 코인 등의 정보가 기록된다.

(b) : 저장했던 데이터를 불러와 적용하는 버튼이다.

(c) : 시작화면으로 돌아가는 버튼이다.

(d) : 환경설정 창을 활성화하는 버튼이다.

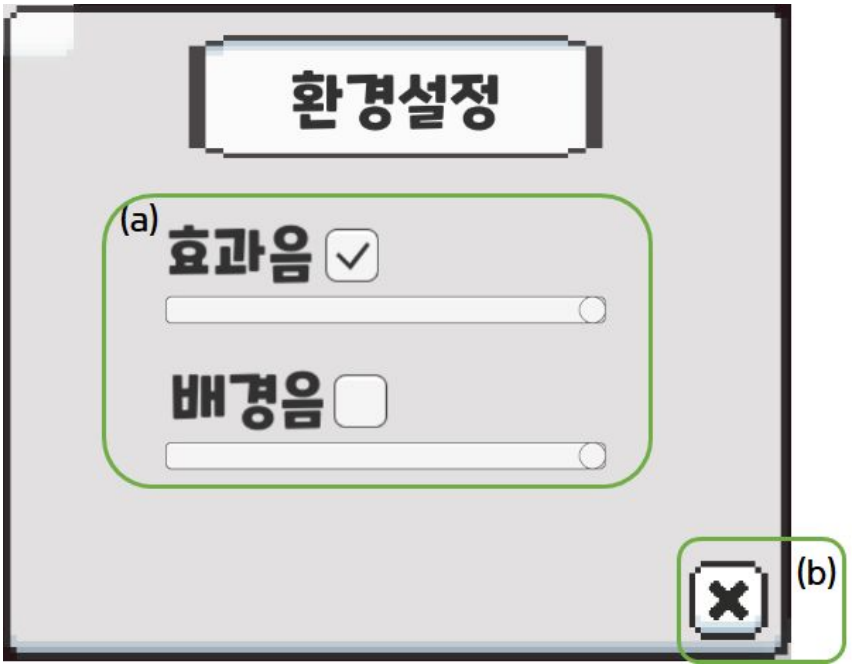
(e) : 지금 열려있는 메뉴 창을 닫는 버튼이다.



[인벤토리 창]

(a) : 플레이어가 소유중인 특수 아이템의 소지 상태를 표시하는 슬롯이다. 특수 아이템은 좌상단 부터 차례대로 들어간다.

(b) : 획득한 무기 순서대로 채워진다.(단, 가득찼을 경우 더 이상 무기가 채워지지 않는다.)

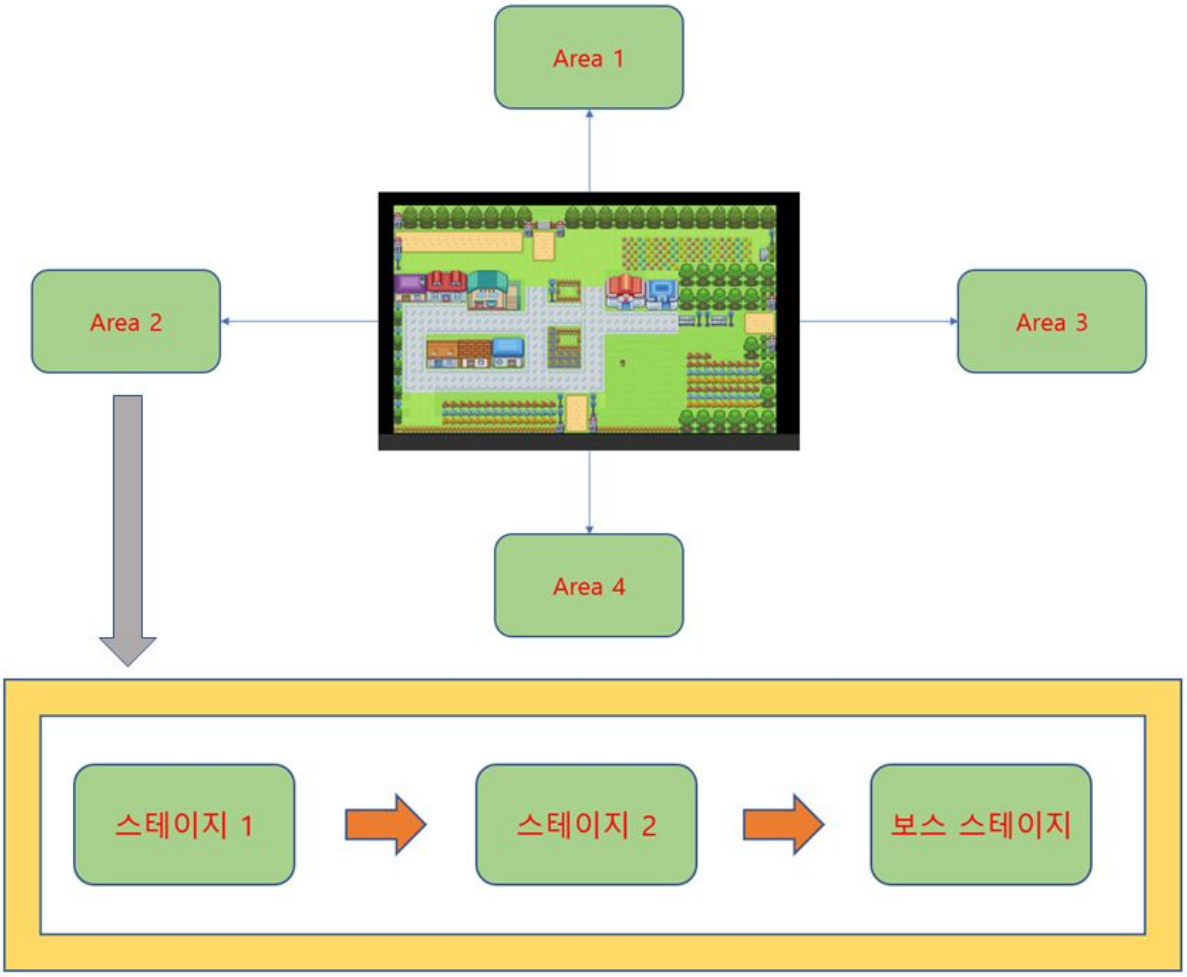


[환경설정 창]

(a) : 효과음과 배경음의 크기를 조절한다.

(b) : 환경설정 창에 나간다.

5) 맵 디자인



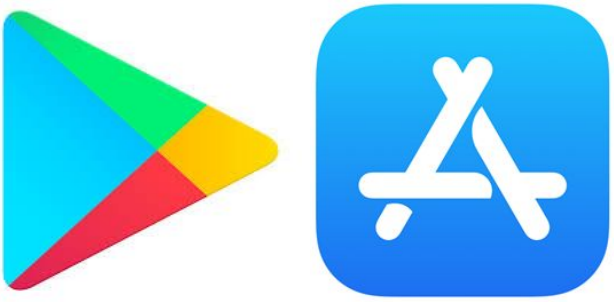
1. 시작 지점 : 게임이 시작되는 곳으로 각 Area를 클리어하거나 플레이어가 죽었을때 다시 시작하는 지점이다.
2. 모든 Area는 2개의 일반 스테이지와 1개의 보스 스테이지로 구성된다.

6) 몬스터

1. 근접 몬스터 : 근접 공격만 하는 몬스터
2. 원거리 몬스터 : 원거리에서 공격만 하는 몬스터
3. 보스 몬스터 : 각각의 스테이지에서 가장 강한 몬스터

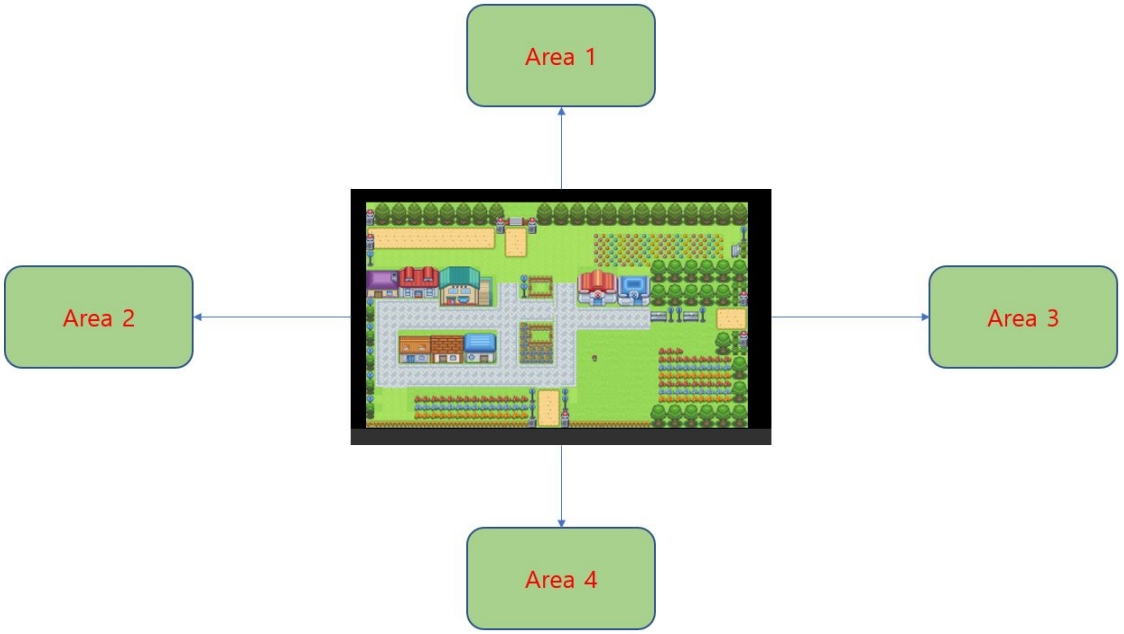
7) 사용자 시나리오

1. 스토어에서 게임파일을 다운 & 설치



2. 게임 시작 버튼 클릭
   - 환경설정(시작전에 설정, 게임 중에도 설정 가능)

3. Area 1,2,3,4을 클리어 한다.

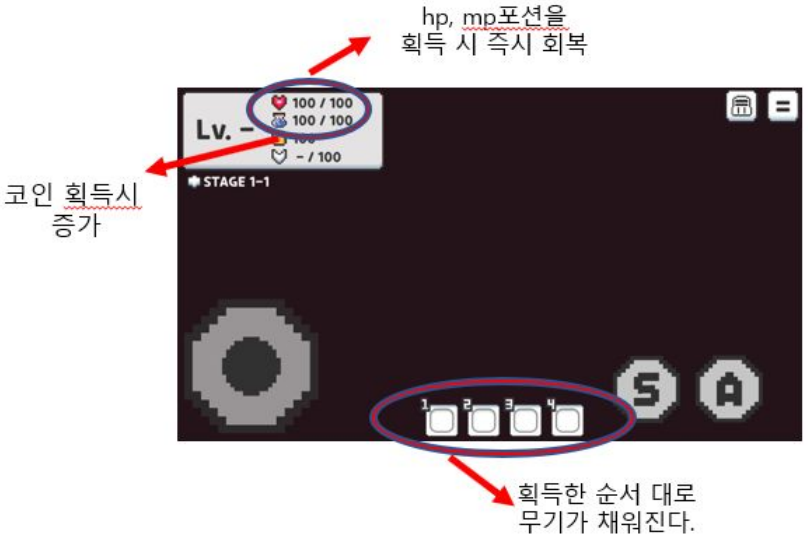


3.1. 게임 중간에 플레이어가 죽으면 home에서 다시 시작한다.

3.2. 게임 중 플레이어 상태가 이상이 있을 경우 hp, mp물약을 획득시 즉시 회복시킨다.(단, 상태가 정상일 경우 획득 되지 않는다.)

3.3. 무기 아이템을 획득기 슬롯창에 순차적으로 채워진다.(슬롯이 다 찼을 경우 무기를 획득할 수 없다.)

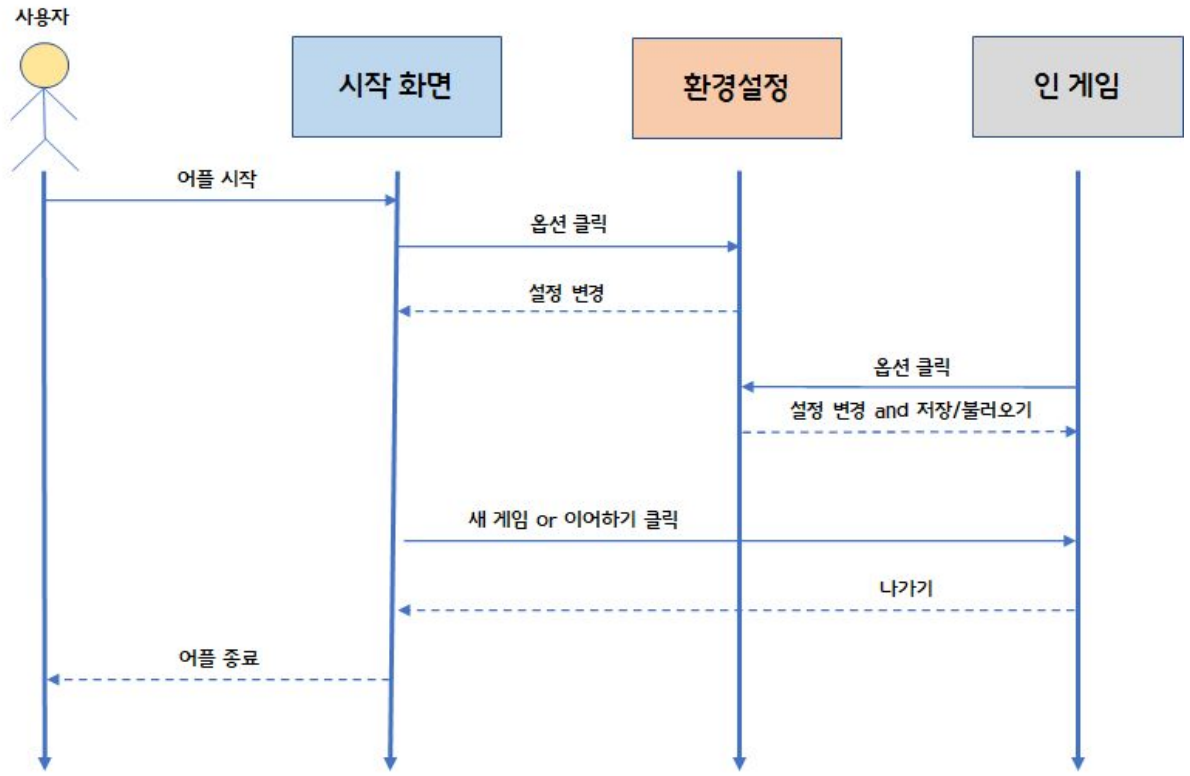
3.4. 코인을 획득시 즉각적으로 재화가 늘어난다.



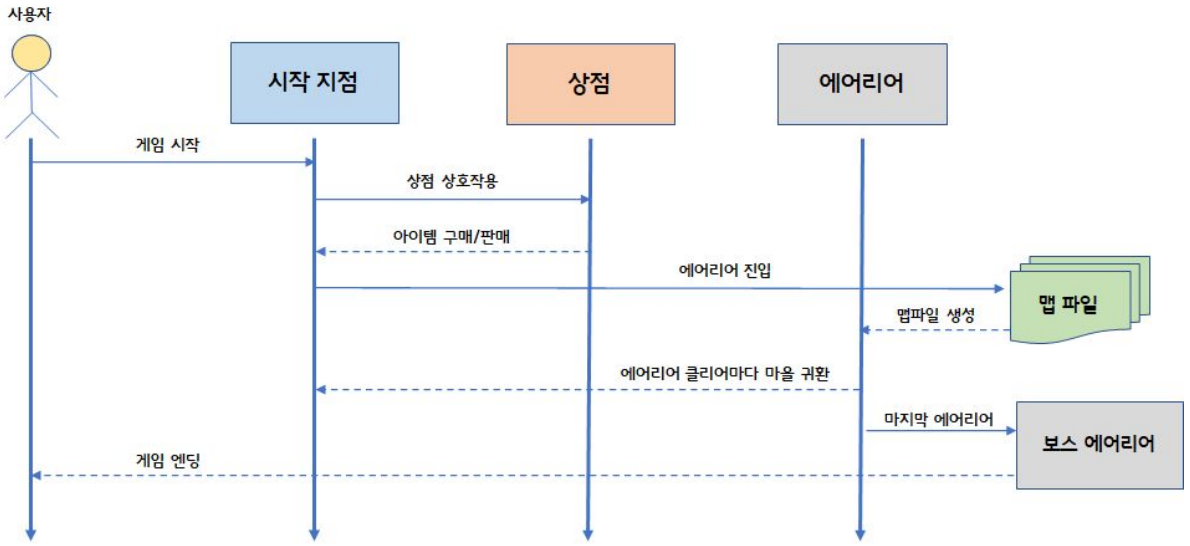
4. 각 Area 클리어 전/후로 상점에서 필요한 아이템 구매

5. 나머지 Area 들을 클리어하기

6. 모든 Area를 클리어하면 게임은 끝난다.



[시퀀스 다이어그램 1 - 시스템 시나리오]



[시퀀스 다이어그램 2 - 인 게임 시나리오]

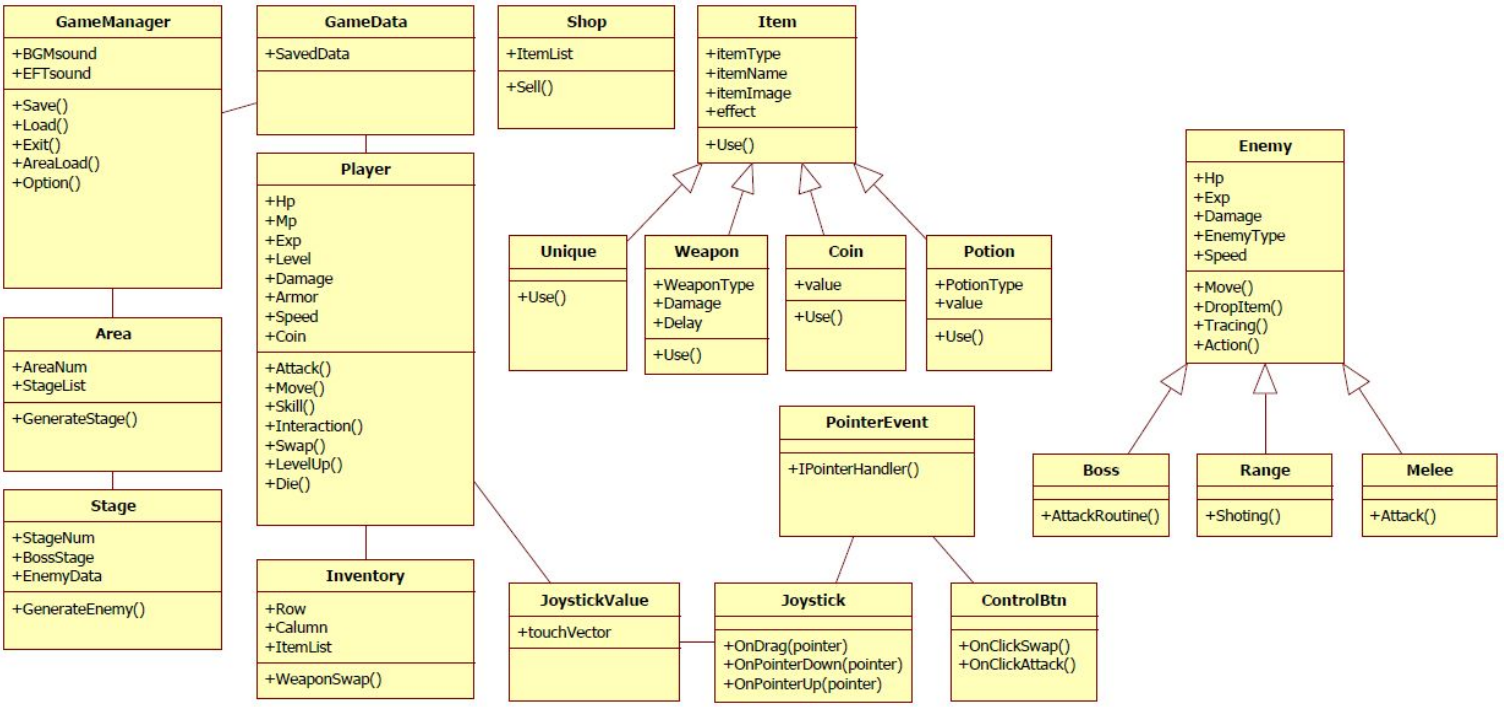
3. 개발자 가이드

3.1 모듈별 상세 명세

- 유니티 내부의 모듈 : 게임 개발 툴인 유니티 내부에는 다양한 모듈이 집합되어 있고 입력, 출력, 이벤트 발생은 다양한 기능을 지원한다.
- 게임 개발 툴인 Unity를 설치할때 추가적으로 설치해야하는 모듈이다.
    1) Window Build Support : 기본적인 윈도우 환경에서의 게임 구동을 위해서 설치해야하는 모듈이다.
    2) Android Build Support : 안드로이드 플랫폼에서 만든 게임이 돌아가로독 환경에 맞춰 게임을 빌드하는 것을 도와주는 모듈.
    3) Mac Build Support : Mac 플랫폼에서 만든 게임이 돌아가도록 환경에 맞춰 게임을 빌드하는 것을 도와주는 모듈**.**

3.2 데이터 명세

1) 클래스 다이어그램

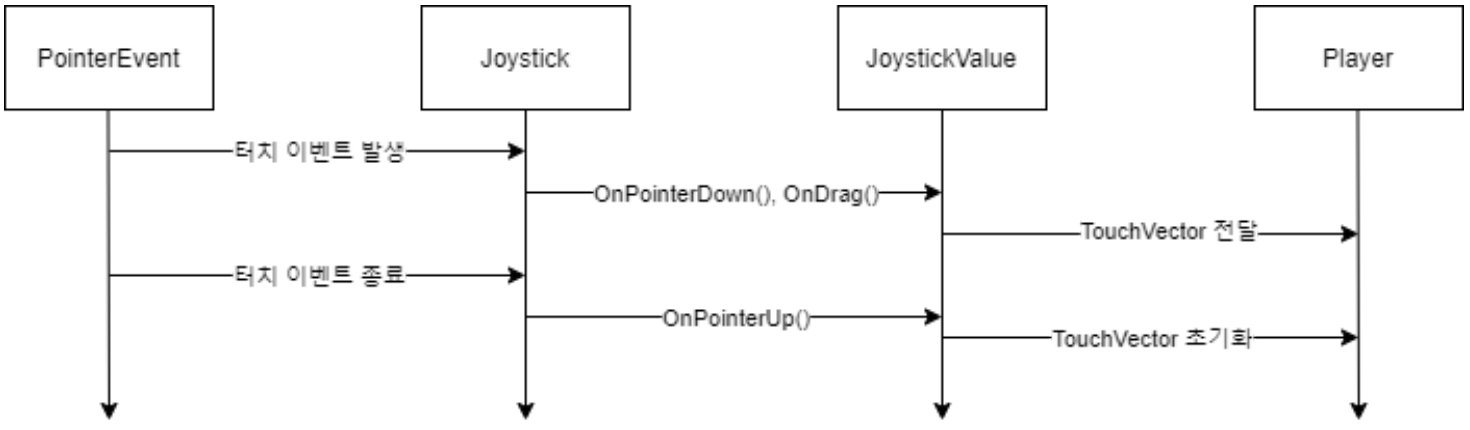


[클래스 다이어그램]

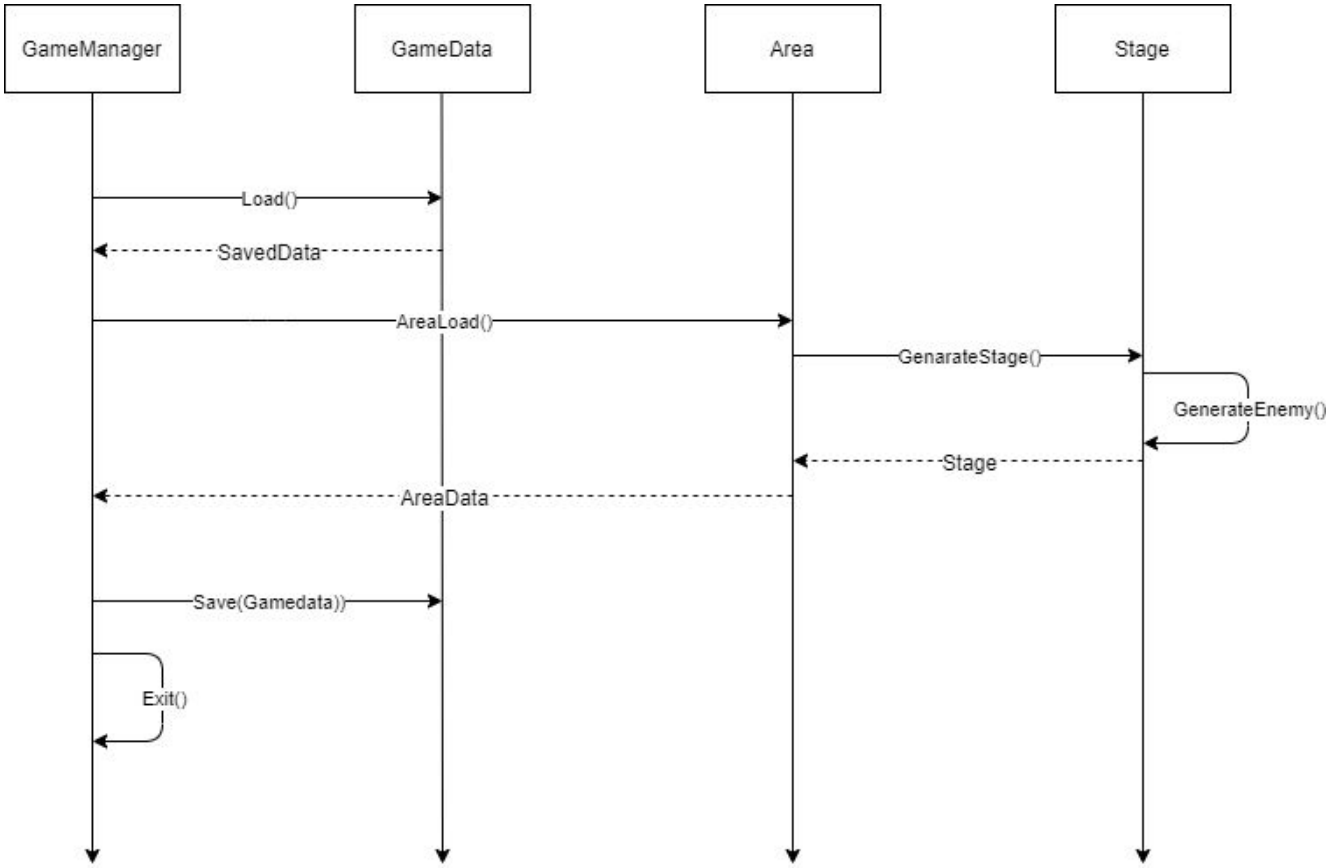
- 클래스 부가설명

- **GameManager :** 게임의 전반적인 진행을 관리하는 클래스로 게임이 실행될때 생성되어 게임이 끝나는 시점까지 파괴되지 않고 게임을 진행시킨다.
- **Area :** 플레이어가 클리어하는 큰 단위의 맵으로 하위에 작은 단위의 맵인 스테이지가 일반 2개, 보스 1개 생성된다.

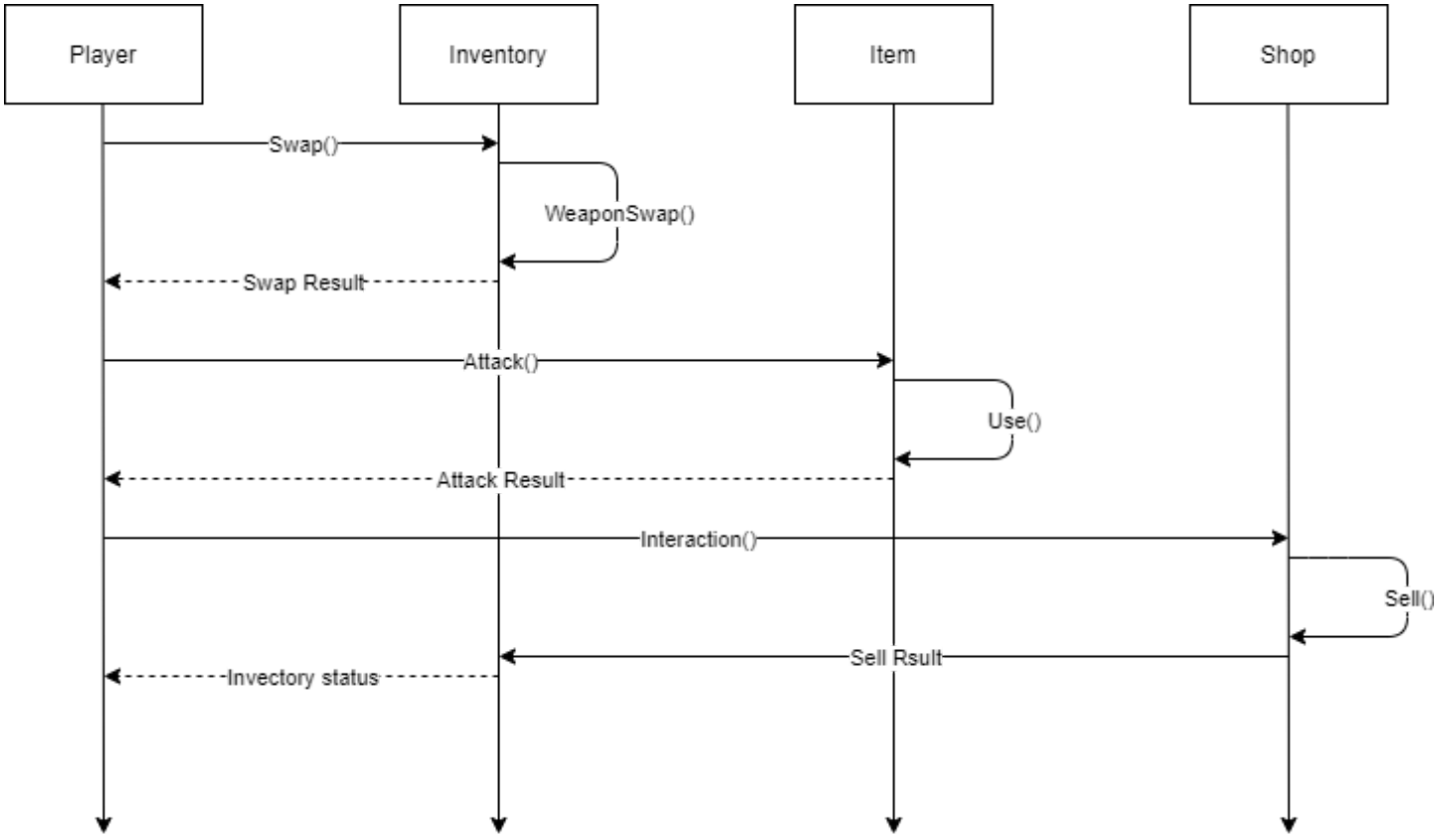
- **Stage :** 플레이어가 클리어하는 작은 단위의 맵으로 생성될때 지정된 위치에 적을 위치 시키게 된다.
- **GameData :** 사용자의 진행 상황, 플레이어 캐릭터의 상태, 재화 상태 등의 정보가 저장되며, GameManager에 의해 조작 된다.
- **Player** : 사용자가 직접 조작하게 되는 객체 클래스로, 이동, 상호작용, 공격 등 플레이어 중심의 기능을 제공한다.
- **Inventory** : 플레이어 내부에 존재하며 습득한 아이템을 처리하는 클래스이다. 현재 가지고 있는 아이템의 정보를 확인 할 수 있으며 무기 소지 현황을 알 수 있다.
- **Shop** : 시작지점에 존재하는 상점을 관리하는 클래스이다. 플레이어가 보유하고 있는 코인을 사용하여 금액을 지불하고 아이템을 구매할 수 있다.
- **Item** : 아이템 구조는 큰 틀의 아이템 타입과 이름, 이미지를 기본으로 두고 기능을 상속하여 하위로 4개의 클래스를 만들게된다.
    - Unique : 고유 아이템으로, 플레이어의 기본 능력을 강화 시키는 아이템 클래스.
    - Weapon : 무기 아이템으로, 플레이어의 공격을 보조하는 아이템 클래스.
    - Coin : 코인 아이템으로, 습득 시 플레이어의 코인 값을 올려주는 아이템 클래스.
    - Potion : 물약 아이템으로, 습득 시 플레이어의 상태를 회복시키는 아이템 클래스.
- **Enemy** : 적 클래스의 구조는 기본적인 아이템 드랍, 이동, 추적 등을 공통으로 하고 기능을 상속하여 하위 클래스 3종류로 나누어 기능을 부여한다.
    - Boss : 보스 타입의 적으로 고유의 행동패턴이 존재한다.
    - Range : 원거리 타입의 적으로 목표대상에게 물체를 발사한다.
    - Melee : 근거리 타입의 적으로 목표대상에게 이동하여 공격을 실행한다.
- **PointEvent** : 화면을 터치하는 것을 통한 각종 이벤트를 실행하는 클래스이다. 관련이 있는 클래스는 플레이어의 조작을 보조하는 **Joystick 클래스**와 공격 및 부가기능을 담당하는 **ControlBtn 클래스**가 있다.
- **JoystickValue** : Joystick 클래스를 통해 계산된 값을 플레이어에게 전달하는 클래스이다.



[시퀀스 다이어그램 1 - 터치이벤트]



[시퀀스 다이어그램 2 - GameManager]



[시퀀스 다이어그램 3 - Player]

3.3 소스 코드 디렉토리 구조(GitHub 링크 포함)

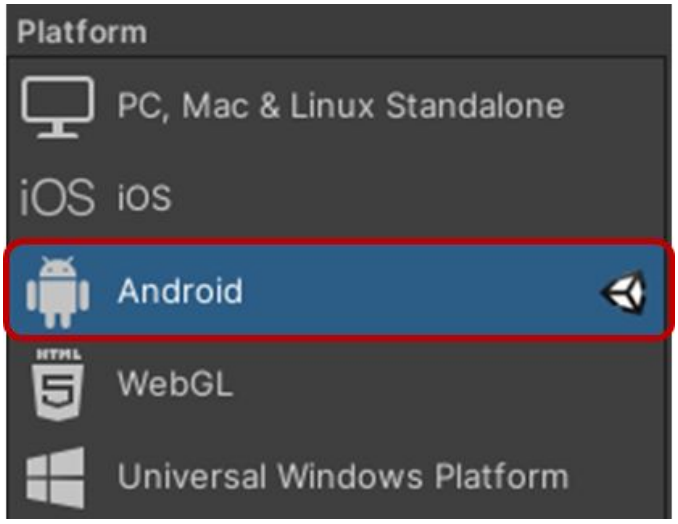
- GitHub link : https://github.com/hyeonjun414/Lucas-Adventure
- 디렉토리 구조

Assets

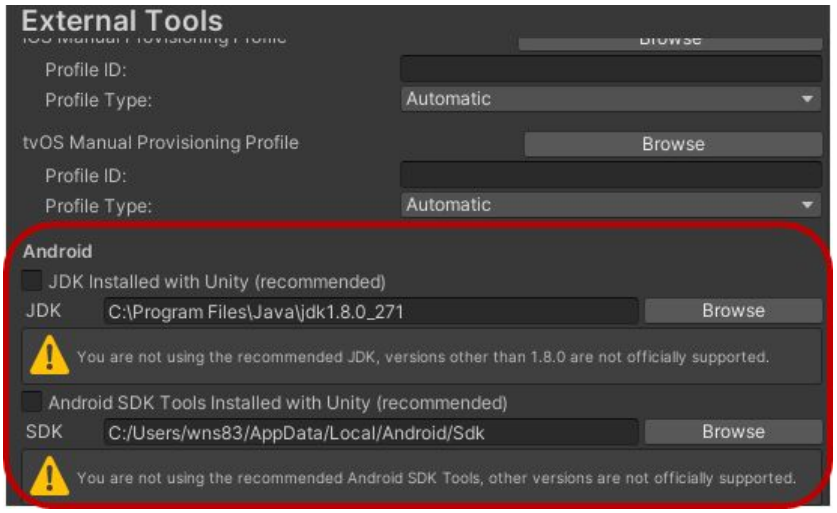
1. Animations : 행동이 있는 객체에 대한 애니메이션을 저장.
    1) Player
    2) Enemy
        a) Normal
        b) Boss
2. Fonts : 게임 내에서 사용되는 폰트를 저장.
3. Resources : 생성한 오브젝트를 재사용가능한 형태로 저장.
    1) Items
    2) UI
    3) PlayerAndEnemy
4. Scenes : 시작화면, 플레이화면등 다양한 장면을 구성.
5. Scripts : 오브젝트가 기능을 수행할수 있도록 하는 스크립트파일을 저장.
    1) Activity
    2) Control
    3) Item
    4) UI
6. Sprites : 오브젝트에 적용되는 이미지를 저장.
    1) Objects
    2) Tiles
7. Maps : 미리 구성된 스테이지의 Scene을 저장.
8. Sounds : 효과음, 배경음에 적용되는 사운드 파일을 저장.
    1) Bgm
    2) Effect

3.4 빌드 방법

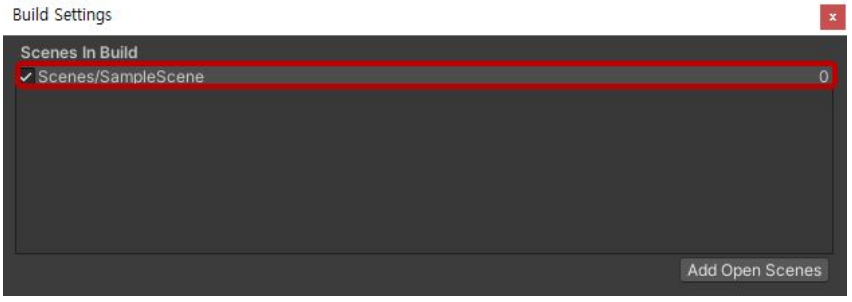
1) 사용중인 개발 툴인 Unity의 Build Setting에서 Mobile Platform으로 전환 한다.



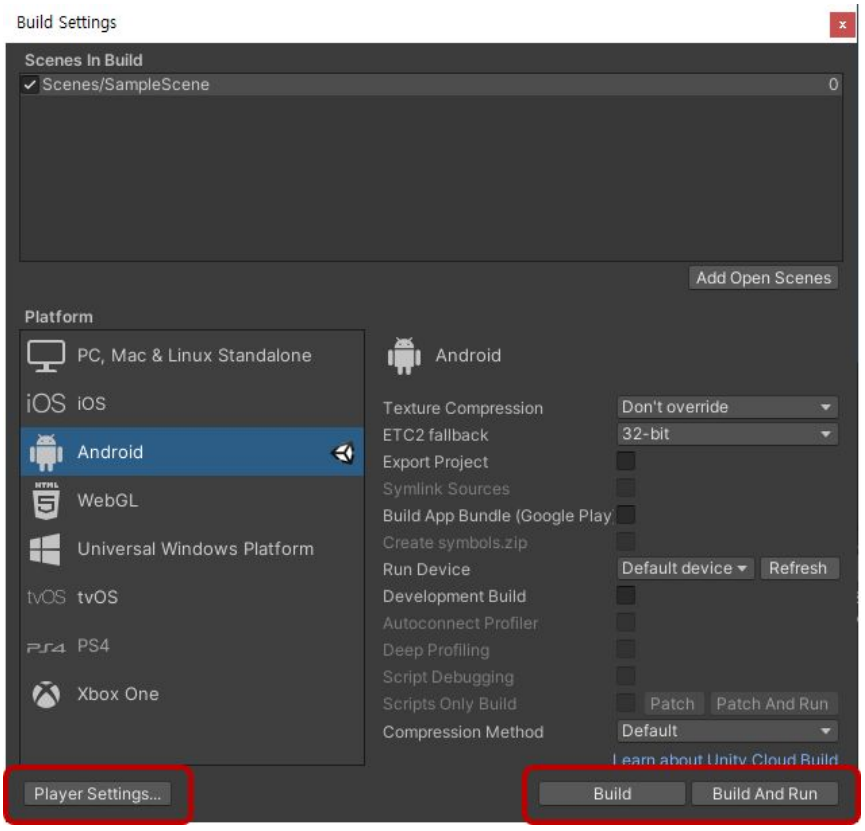
2) Preferences의 External Tools에서 모바일 환경 구성에 필요한 JDK, SDK를 버전에 맞게 설치해준다.



3) Build Setting에 실행에 사용되는 Scene을 추가해준다.



4) **Player Setting에서 어플의 이름, 아이콘등을 수정해준뒤 Build 또는 Build and Run 버튼을 클릭한다.**



3.5 테스트 방법

- **각자가 만든 요소들을 자체 테스트를 진행한다.**
- 각 주차의 금~일요일에 zoom/대면을 통해 서로가 만든 요소를 합쳐서 테스트를 실행한다. 이때 각자가 만든 요소를 합치는 과정에서 생긴 오류를 함께 해결한다.

4. 테스트 계획

[1차 테스트]

테스트 목록	테스트 내용	성공 판정 기준
1	아이템 수집, 몬스터 타격	-아이템을 수집시 인벤토리에 저장 -몬스터를 타격시 몬스터의 hp가 떨어져서 최종적으로 죽게 되면 플레이어의 exp가 증가 되는지 확인
2	맵과 플레이어 연동	-플레이어가 맵 안에서만 움직이는지 확인 -플레이어가 사물이나 벽과 충돌시 더 이상 이동이 되지 않는지 확인
3	몬스터 활동 자동화	-몬스터가 스스로 움직이는지 확인 -몬스터 활동범위안에 플레이어가 오면 플레이어를 공격하는지 확인.
4	맵 불러오기	-각각의 에어리어를 실행시 2개의 스테이지와 정해진 보스 스테이지가 나오는지 확인
5	전투 중 스킬 사용확인	-스킬 버튼을 작동시 작동 여부 확인 -스킬이 몬스터에게 타격시 몬스터의 hp가 떨어지는지 확인

[2차 테스트]

테스트 목록	테스트 내용	성공 판정 기준
6	레벨업으로 인한 플레이어 능력치 향상	-플레이어가 레벨업시 능력치가 변화되는지 확인
7	보스 공격 패턴	-보스가 플레이어를 추적하여 공격 패턴을 수행하는 지 확인 - 패턴이 정해진 로직에 따라 움직이는지 확인
8	모바일 환경 적용	-폰에서 게임이 진행되는지 확인 -가상 패드와 가상 버튼으로 게임과 상호작용이 뜻대로 되는지 확인
9	저장/불러오기	-저장한 후 게임을 종료하고 다시 실행시 제대로 불러오는 지 확인

10	상점에서 아이템 구매시 인벤토리 창에 이동 확인	-상점에서 구매된 아이템이 인벤토리에 들어가는 지 확인 -아이템 구매 시 일정한 양의 코인이 사용되는지 확인
11	background-music 작동	-무기를 사용시 타격음이 제대로 나오는지 확인 -스테이지에 맞는 bgm이 나오는지 확인 -플레이어가 사망할때 사망bgm이 나오는지 확인 -스테이지를 클리어시 bgm이 나오는지 확인

5. 개발일정 및 역할분담

5.1 개발일정

개발 일정	내용
1 ~ 3주차	팀 구성 및 프로젝트 초안, 정식 제안서 작성
4주차	게임 구성에 필요한 음향-그래픽-맵디자인 검색 Unity에서 개발에 도움되는 에셋 검색 및 학습 Unity게임 환경에서의 이미지 표현 방식에 대한 토의 및 결정 Github를 통한 통합 협업환경 구축
5주차	그래픽을 구성할 2D이미지 검색 및 선택 중앙 스타팅포인트 맵 설계 및 제작 플레이어블 캐릭터와 아이템 적용방식에 대한 설계 스타일에 맞는 도트이미지 생성 및 수정
6주차	3~4가지 테마의 맵 설계 및 제작 맵 생성 방식에 대한 설계 및 구현, 아이템의 설계 장착아이템과 소비아이템의 사용 방식과 적용에 대한 구현

7주차 (1차 테스트)	플레이어 캐릭터의 제작과 몬스터와의 상호작용 구현 각 오브젝트(몬스터, 플레이어)의 애니메이션 구현 공격(타격, 피격)의 이펙트(파티클, 애니메이션)와 음향 적용 플레이어 스킬 구현 1차 테스트 실행(테스트목록 1~5)
8주차	스테이지 구현 시스템 설계 및 구현 몬스터의 생성방식과 스테이지 클리어 보상구현 다음 스테이지로의 전개 방식 구현
9~10주차 (2차 테스트)	보스 및 스테이지 구현, 상점 구현 화면을 구성하는 인터페이스UI제작 모바일 환경에 맞춘 조작방식과 UI수정 저장-불러오기 기능 구현 2차 테스트 실행(테스트목록 6~11)
11~12주차	테스트 플레이 및 버그 수정, 테스트에 따른 난이도 수정
13~14주차	사용자 가이드 및 개발자 가이드 작성, 문서 최종 수정
15주차	발표

5.2 역할 분배표

이건희	장현준
캐릭터&몬스터 레벨 디자인	그래픽
무기	상호작용 이벤트
맵제작	스토리
음향	조작

5.3 협업 방식

- Github를 통한 프로젝트 공유
- 일단 위에 주어진 대로 작업을 처리하되, 필요에 의해 한 가지 작업을 동시에 진행할 수 있다
- 프로젝트 완성단계에서 서로 완성한 요소들을 적절히 조합하고 부족한 점을 보완 및 버그수정을 한다.
- 각 주차의 금~일요일에 zoom/대면을 통해 서로가 만든 요소를 합쳐서 테스트를 실행한다.

6. 위험 및 프로젝트 관리

6.1 위험관리

1) 위험 계획 수립

- 코드의 손실을 방지하기위해 Github을 사용하여 인터넷상에 백업시키며 프로젝트의 안전성을 확보한다.
- 매주 2회정도 정기적으로 회의시간을 가진다.

2) 위험 식별

- 에러로 인해 Github의 코드가 손실되면 어떻게 할 것인가?
- 정해진 기간 안에 하고자 했던 기능을 전부 추가할수 있는가?
- 구현한 기능이 원하지 않는 결과를 수행한다면 어떻게 할 것인가?

3) 위험 대응

- 코드의 손실을 방지하기 위하여 GitHub외에도 클라우드 환경에 코드를 백업시켜놓는다.
- 되도록 정해진 개발일정에 맞도록 진행하지만, 만약 일정에 맞추지 못한다면 완성도에 초점을 두고 중요성이 낮은 기능을 우선적으로 조정한다.
- 코드의 에러가 일어나지않도록 독립적인 개발공간에서 각 기능을 추가하며 한 기능이 완성되면 팀원들이 같은 내용을 테스트하고, 테스트에 통과하면 통합 개발환경으로 병합한다.

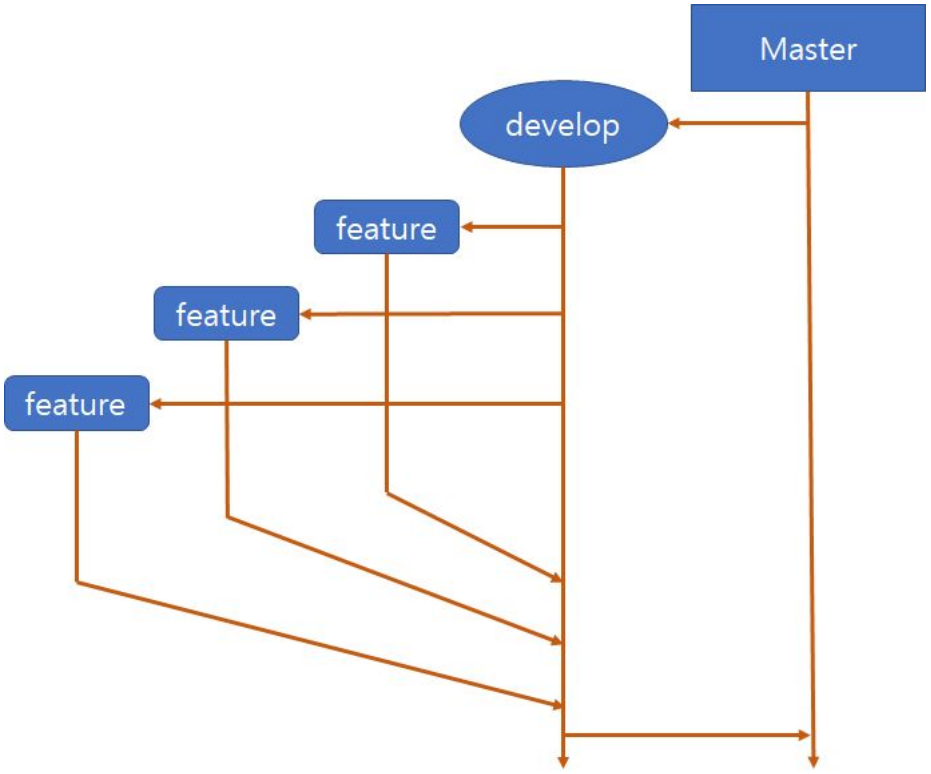
4) 위험 통제

- 회의시간마다 기능구현의 진행정도를 공유하고, 진행정도에 맞춰 일정 리스트를 재조정한다.
- 주에 한번씩 프로젝트 위험도를 측정하고 필요에 따라 위험 리스트를 수정한다.

6.2 프로젝트 관리

- Github link : https://github.com/hyeonjun414/Lucas-Adventure

프로젝트는 깃허브를 통해 코드를 공유하며, 아래 이미지의 개발 형태를 갖추고 있다.



- master에서 develop branch로 나누어지고 develop에서 각 기능에 맞게 나누어 각각의 feature branch로 다시 나뉘게 된다.
- develop branch는 게임이 완성되도록 완성된 feature branch부터 순서대로 병합시키고 모든 feature가 병합이 완료되면 develop은 정상적인 가동이 가능하다고 판단하여 master에 완성된 게임 버전 1을 올리게 된다.
- 가능한 master에는 파일을 업로드 하지 않으며 모든 기능을 갖춘 develop branch만 올라가게 된다.

7. 참고 문헌 및 용어 해설

7.1 용어 해설

- 로그라이크 : Rogue-like라는 의미로 Rogue라는 게임의 특징이 하나의 게임 장르가 되었다. 간단히 말하면 세이브가 없는 어려운 턴제 RPG특징을 지니고, 방과 통로로 이루어진 스테이지. 게임 내 요소의 무작위 출현등의 특징을 가지고 있다.

- RPG : Roll Playing Game의 약어로, 시작은 역할을 정해 모험을 헤쳐나가는 테이블 보드게임에서 시작되었다. 모험을 통한 플레이어의 성장에 초점을 맞춘 게임이다.

- 탑-다운 뷰 : 플레이어의 시점이 위에서 아래로 내려다보는듯한 시점을 뜻한다.

- Hp, Mp, Exp : Health Point, Magic Point, Experience Point의 약어로 체력과, 기술을 사용하기 위한 자원인 마나, 그리고 경험치를 의미한다.

- JDK : Java Development Kit의 약어로, 유니티 게임 빌드에 사용된다.

- SDK : Software Development Kit의 약어로,유니티 게임 빌드에 사용된다.