AD 프로젝트 보고서

20191555 김규민 20191566 김유빈

프로젝트 주제 선정의 동기:

저희가 수행한 AD프로젝트의 주제는 "파이어 보이와 워터걸"이라는 플랫포머 게임입니다. 이러한 주제를 선정하게 된 이유는 첫째, 저희가 관심이 있는 것으로 과제를 수행하기 위함입니다. 둘째는 게임은 수많은 요소들이 상호작용하는 소프트웨어이므로 이번 학기 여러 과목에서 배웠던 "객체 지향 프로그래밍"의 좋은 연습이 될 것 같아서입니다. 셋째, 파이썬에는 게임을 만들기 편하게 해주는, pygame등의 모듈이 있어 보다 편하게 게임을 만들 수 있지만, PyQt를 이용해 게임 소프트웨어를 밑바닥부터 설계하면 소프트웨어 제작 공부에 도움이 될 것 같아서입니다.

프로젝트 요구사항

1. 기능적 요구사항
  1-1. 2명의 플레이어가 각각의 조종키로 동시에 움직일 수 있어야 한다.
  1-2. 캐릭터가 화면 밖을 나갈수 없다.
  1-3. 스테이지에는 중력이 작용하여 플레이어가 움직이는데 제약이 있다.
  1-4. 서있을 수 있는 땅이 있어야 한다.
  1-5. 지나갈 수 없는 벽이 있어야 한다.
  1-6. 불을 플레이하는 플레이어는 물 웅덩이에 닿으면 죽고,
  물을 플레이하는 플레이어는 불 웅덩이에 닿으면 죽는다.
  1-7. 어떤 플레이어든 독에 닿으면 죽는다.
  1-8. 버튼은 누르면 지나갈 수 있는 장애물이 있어야한다.
  1-9. 두 캐릭터 모두 캐릭터에 맞는 문에 서면 클리어가 된다.
  1-10. 부드러운 움직임을 위해 캐릭터가 가진 가속도에 따라 움직이게 한다.
  1-11. 캐릭터가 움직일 때 애니메이션이 있어야한다.
  1-12. 맵의 추가 제작이 쉬워야한다.

2. 사용자 인터페이스 요구 사항
  2-1. 처음 시작할 때 바로 게임으로 들어가는 게 아닌 타이틀 화면 배치
  2-2. 스테이지 클리어 시 다음 스테이지로 넘어갈 때 로딩화면 삽입.

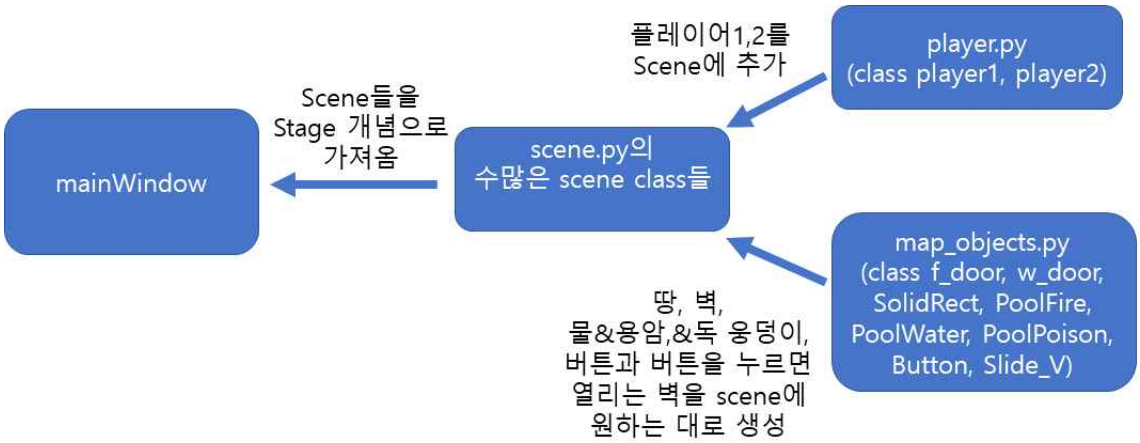
3. 비기능적 요구사항
  3-1. 조금 더 어려운 길이 되더라도, PyQt5를 최대한 활용하여 제작한다.
  3-2. 자연스러운 화면을 위해 60fps로 게임을 구동하며, 렉이 발생하면 안된다.

소프트웨어 구조설계

모듈	클래스	역할
main_screen.py	Mainwindow	게임을 실행하고 모든 것을 불러오는 화면
scene.py	Title	게임을 시작할 때 플레이어에게 보여주는 타이틀

	Scene0 ~ Scene3	플레이어가 클리어해야 할 스테이지들
	End	모든 스테이지 클리어 시 나올 화면.
player.py	Player1	불 캐릭터 형태, 방향키로 이동하는 플레이어 객체
	Player2	물 캐릭터 형태, W,A,D로 이동하는 플레이어 객체
map_object.py	f_door	스테이지 클리어를 위해 불 캐릭터가 도착해야 할 목적지.
	w_door	스테이지 클리어를 위해 물 캐릭터가 도착해야 할 목적지.
	Solidrect	지나갈 수 없고, 밟고 올라 설 수 있는 사물
	Poolfire	불 웅덩이, 물 캐릭터가 닿으면 그 캐릭터는 죽음.
	PoolWater	물 웅덩이, 불 캐릭터가 닿으면 그 캐릭터는 죽음.
	PoolPoison	독 웅덩이, 어떤 캐릭터가 닿든 그 캐릭터는 죽음
	Button	버튼, 연결되어있는 사물을 작동시킴.
	Slide_V	지나갈 수 없는 벽, 연결된 버튼이 눌러지면 위아래로 움직여 길을 틈.

소프트웨어의 구조도



소프트웨어 상세설계

모듈	클래스	메서드	기능
main_screen.py	Mainwindow	timerEvent	타이머 이벤트 핸들러. 지정한 시간(16ms)마다 changeScene 메서드 실행.
		changeScene	현재 스테이지가 클리어되었다면, 화면에 표시되는 스테이지를 다음 스테이지로 넘김
scene.py	Title	timerEvent	지정된 시간마다, 자신의 keys_pressed 집합의 길이를 체크, 0이 아니면 스테이지 클리어로 취급.
		keyPressEvent	키가 눌러졌을 때 keys_pressed 집합에 눌려진 키를 추가.
		keyReleaseEvent	키를 떼었을 때 keys_pressed 집합에서 뗀 키를 제거.
	Scene0 ~ Scene3,	spawn	scene에 파이어보이, 워터걸 생성 및 시작점 좌표 지정.
		terrain_detect	scene 안의 플레이어 객체들의

	End		ground_detect 메서드 실행.
		object_update	scene 안에 있는 모든 객체들의 상태 갱신 메서드 실행.
		death	플레이어가 scene에서 보이지 않는 경우, spawn 메서드 실행.
		stage_clear_detect	두 플레이어가 목적지에 도착했는지 감지, 둘 다 도착했으면 스테이지 클리어로 취급.
		keyPressEvent	키가 눌러졌을 때 그 키를 각 플레이어 객체들의 keys_pressed 집합에 추가
		keyReleaseEvent	키가 떼어졌을 때 그 키를 각 플레이어 객체들의 keys_pressed 집합에서 제거
		game_update	scene에 있는 모든 객체들의 상태를 갱신시킴.
		timerEvent	타이머 이벤트 핸들러. game_update 메서드를 지정한 시간(16ms)마다 실행하며, 갱신된 모든 정보를 현재 화면에 업데이트시킴.
player.py	Player1, Player2	key_in	플레이어 객체의 keys_pressed 집합에 좌우키(혹은 A, D 키)가 들어가 있으면 해당 방향에 맞는 값의 가속도를 플레이어 객체에게 부여한다. 플레이어가 바닥에 서있을 때 상단 방향키(혹은 W 키)가 눌러졌을 경우 jumped 값을 True로 지정한다.
		gravity	플레이어 객체가 중력가속도 가중의 형태로 중력을 받게 한다.
		ground_detect	플레이어 객체가 map_object중 Solidrect 와 충돌하여 못 지나가게 하고, Solidrect를 밟고 서 있을 수 있게 한다.
		jump	플레이어 객체의 jumped 가 True일 경우 작동하며, 플레이어 객체가 바닥과 수직인 방향으로 가속도를 받게 하여 점프하게 한다.
		move_per_frame	플레이어 객체가 가지고 있는 횡 방향, 종 방향 가속도에 비례하여 scene 안에 있는 해당 객체를 이동.
		inertia	플레이어 객체의 횡 방향 가속도의 절댓값이 0에 가까워 질 수 있도록 관성을 줌..
		char_animate	플레이어가 움직일 때 부드러운 애니메이션을 가지기 위해 플레이어 캐릭터의 사진을 계속해서 새로운

			사진으로 변경.
		player_update	char_animate, key_in 등 플레이어 객체의 상태 갱신용 메서드들을 모두 실행
map_object.py	f_door	open	플레이어 객체 fire(혹은 water)와 접촉하면 변수 opened를 True로 설정하고 열려져있는 상태로 그림이 변경됨.
	w_door		
	Solidrect		
	Poolfire	kill_water	플레이어 객체 water이 닿으면 water을 화면에서 숨김.
	PoolWater	kill_fire	플레이어 객체 fire이 닿으면 fire을 화면에서 숨김.
	PoolPoison	kill_all	플레이어 객체가 닿으면 접촉한 객체를 화면에서 숨김.
	Button	push_detect	플레이어 객체가 해당 버튼을 누르고 있는지 아닌지를 감지. 누르고 있으면 변수 pushed를 True로 설정하고 화면에서 숨김, 안누르고 있으면 pushed는 False로 설정되고 화면에 나타남.
	Slide_V	collide	플레이어가 Slide_V 객체에 부딪히고, 통과할 수 없게 하는 메서드.
		slide	해당 객체와 연결된 버튼의 pushed값이 True인 경우, 정해놓은 방향으로 움직여 길을 틈.

코드 및 사용된 이미지 파일은 git에 포함됨.

프로젝트 진행 후기

PyQt5를 이용해 게임을 만들며, 게임 제작용 툴이 아무것도 없는 상황에서 게임을 만드는 것이 생각보다 고려해야 하는게 많음을 느끼게 되었다. 캐릭터가 키에 따라 움직이고, 벽에 부딪히면 못 지나가고, 자연스럽게 점프하고... 이 모든 것들을 직접 넣는 것은 매우 번거로운 일이었으나, 한편으로는 그런 작업들을 통해 게임을 구성할 때 제작자가 생각해야 하는 모든 요소를 생각해보며 게임 소프트웨어 제작에 대한 이해도가 상승하였다.

또한, 이번 학기가 시작할 때 소프트웨어에서의 '객체'가 무엇인지도 모르던 상태에서 수업들을 통해 배운 '객체 지향적 프로그래밍'을 실천해보려 노력하면서, 수많은 class들이 깔끔하게 정리될 수 있다는 점이 인상적이었다. 다만 이번 프로젝트를 진행할 때, 게임에 들어갈 객체들에 대한 상세한 분석이 부족하여 코드들의 상속 구조 및 상황 처리가 일관성이 없었고, 결과물이 잘 작동하긴 하지만 코드가 스파게티가 되버린 느낌이다. 객체들의 관계를 조금 더 면밀히 분석한 상태에서 프로젝트를 진행했다면 더 일관성 있고, 더 아름다운 코드가 작성되었을 것 같아 아쉬움이 남는다.