

2021-1 OSSP

중간보고서

8조 3인분

<https://github.com/CSID-DGU/2021-1-OSSPC-3People-8>

2015110510 통계학과 김태환

2015112583 산업시스템공학과 박재민

2018111711 의생명공학과 배유진

진행 상황 (일정)

2주차 진행 계획

	1주차	2주차	3주차	4주차	5주차	6주차	7주차	8주차
⇒ 메인 코드 분석 및 발표 준비(모두)	■							
⇒ UI개선, 마우스로 메뉴선택 (박재민) 아이템 추가(배유진) 창 크기 조절(김태환)		■						
⇒ 코인, 라이프 시스템(박재민) 적에 따라 score다르게 (배유진)			■					
⇒ 서버 기능, 랭킹 저장 시스템(김태환)			■	■				
⇒ 미사일 공격하는 적 추가 (박재민) 구체적인 업적 시스템 완성(배유진)				■				
⇒ 메뉴 화면 추가(박재민) 아이템 추가(배유진) 업적 시스템 추가(김태환)					■			
⇒ PvP모드, 협동 모드(모두)						■	■	
⇒ 기능 마무리 및 최종 발표준비(모두)							■	■

- 3주차 까지의 기능을 기존 일정대로 수행함

- 2주차 구현 내용 : 메뉴 UI 개선, 마우스 클릭 버튼 생성(다른 모드로 이동하는 버튼), 더블미사일 아이템 추가, 창 크기 조절, 게임 내 이미지 사이즈 조절

- 3주차 구현 내용 : 게임 일시정지 및 라이프 시스템 구현, 서버 기능(로그인, 하이스코어) 구현, 적마다 score 다르게 변경

4주차 진행 계획 조정

	4주차	5주차	6주차	7주차
김태환	서버 기능 마무리	업적시스템(2)		기능마무리 & 디버그
박재민	코인시스템	PVP모드		
배유진	업적시스템(1)	협동모드		

- 4주차 회의에서 진행 계획을 조정하기로 결정(개발 도중 버그 발생으로 일정 지연, 주간보고 피드백 반영, 기획 변경)

- PvP모드, 협동모드 개발 일정을 앞당기고, 4주차 진행 일정을 일부 수정함

- 코인 시스템이 비교적 많은 개발 시간이 소요되는 것으로 판단(웨이브 중 일시정지, 게임 중 코인 아이템 드랍, 코인 메뉴 구현, 코인으로 산 아이템 적용 등), 4주차에 추가적으로 구현하기로 결정


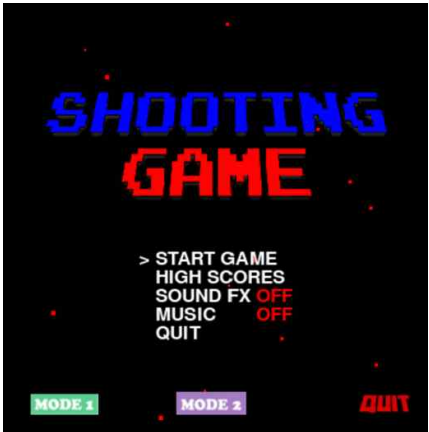
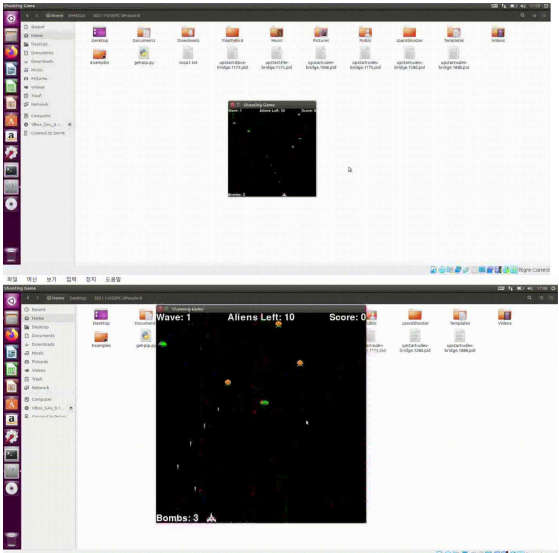
- 서버 최소 기능 구현이 3주차에 완료되어 서버 보안 기능(https 인증, 비밀번호 솔팅 해싱) 추가 시도 -> https 인증 오류로 서버 기능 추가를 포기하고 업적 시스템을 기존 일정보다 앞당겨 구현
- 업적 시스템을 만들고, 업적 페이지를 생성함, 추가로 언어 전환(한글화) 기능 구현


현재 진행 상황 및 계획

- 5주차 까지 구현, 6,7주차에 팀원 모두 PvP 모드와 협동 모드 개발에 주력

	2주차	3주차	4주차	5주차	6주차	7주차
UI	메뉴 UI 개선	로그인 메뉴	언어 선택 기능 추가	하이스코어에서 개인 업적 확인	PvP, 협동모드 UI 수정	협동모드, PvP 모드 기록 확인
	마우스 클릭 가능한 버튼		업적 메뉴 구현	언어 변경 구현 완료		
	창 크기 조절					
인게임	추가 아이템(더블미사일)	라이프 시스템	코인 아이템 드랍	1인 모드 전체적인 코드 수정	게임 화면 두개 동시 생성(PvP)	협동모드 서로 적용되는 쉴드
	게임 이미지 사이즈 조절	일시정지 기능	웨이브 중간 코인 아이템 구매	전체적인 버그 수정	비행기 객체 두개 동시 생성	PvP 적 방해 아이템
		적마다 스코어 다르게			PvP모드 마우스로 게임 화면 넘기는 기능	
서버		로그인, 최고기록 api	업적, 언어 api	하이스코어 업적 api	협동모드, PvP 관련 기능 api	모든 부분 디버깅 및 마무리

진행 상황 (세부 기능)

항목	변경 전	변경 후
2주차		
인터페이스	<ul style="list-style-type: none"> - 파란색 글씨가 검은 배경에서 잘 보이지 않음 - 로고와 메인메뉴만 구현되어 있으며, 마우스로 메뉴 선택 불가능 	<ul style="list-style-type: none"> - 흰색 글씨로 바꾸어 가독성이 향상됨 - 메뉴 화면 밑에 마우스로 선택 가능한 MODE1, MODE2, QUIT 버튼을 추가, 버튼에 마우스를 올리면 버튼 크기가 변하도록 설정 
창 크기	<ul style="list-style-type: none"> - 500*500 사이즈를 기준으로 위치가 상수 값으로 고정되어 있으며, 창 크기를 조절할 수 없음 <pre># size stars background = pygame.Surface((500, 2000)) background = background.convert() background.fill((0, 0, 0)) backgroundLoc = 1500 finalStars = deque() for y in range(0, 1500, 30): size = random.randint(2, 5) x = random.randint(0, 500 - size)</pre>	<ul style="list-style-type: none"> - 모든 값이 화면 크기에 비례하도록 수정하고, shooting_game.py와 sprites.py의 상위 모듈이 있는 main.py를 생성 - 메뉴 선택 화면에서 마우스로 창 크기를 조절 가능하며, 화면 크기에 비례하여 모든 이미지가 조절되기 때문에 기능적인 변화 없이 화면 사이즈를 늘릴 수 있음 

<p>게임 중 PAUSE</p>	<p>- 게임 중에 일시정지 기능이 없고 한 번 게임을 실행하면 게임이 끝날 때 까지 나갈 수 없음</p>	<p>- 키보드로 p버튼을 눌렀을 때 게임이 정지되며 pause화면이 나옴</p>  <p>- 게임이 실행되고 있는 도중 즉, ship객체가 alive 가 true인 상황에서 pause버튼, 키보드p버튼,을 누르면 새로운 while문을 생성해 pause 화면이 나올 수 있도록 코드 작성</p> <pre data-bbox="933 947 1404 1422"># pause -일시정지 elif (event.type == pygame.KEYDOWN and event.key == pygame.K_p): inPmenu = True if id != '': menuDict = {1: restartPos, 2: hiScorePos, 3: fxPos, 4: musicPos, 5: achie else: menuDict = {1: restartPos, 2: hiScorePos, 3: fxPos, 4: musicPos, 5: quiti selectPos = selectText.get_rect(topright-restartPos.topleft) while inPmenu: clock.tick(clockTime) screen.blit(background, (0, 0), area=pygame.Rect(0, backgroundLoc, scr_size, scr_size)) backgroundLoc += speed if backgroundLoc + speed <= speed: backgroundLoc = scr_size*3 ### for event in pygame.event.get(): if (event.type == pygame.QUIT): pygame.quit() sys.exit() elif (event.type == pygame.KEYDOWN and event.key == pygame.K_RETURN): # K_RETURN은 enter누르면 if showHiScores: showHiScores = False elif showAchievement : showAchievement = False elif selection == 1: inPmenu = False break</pre>
-------------------	---	---

3주차

로그인

- 로컬에서만 게임을 플레이하고, sqlite 데이터베이스를 이용해 개인의 하이스코어를 저장하는 것만 가능

database.py

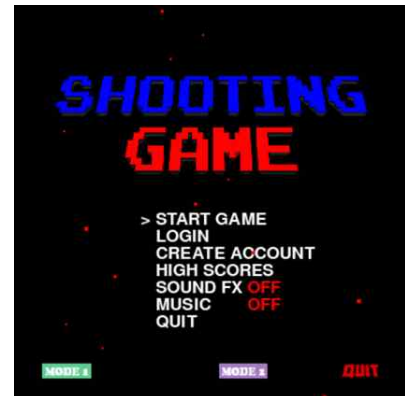
```
@staticmethod
def setScore(hiScores, entry):
    conn = sqlite3.connect(Database.path)
    c = conn.cursor()
    if len(hiScores) == Database.numScores:
        lowScoreName = hiScores[-1][0]
        lowScore = hiScores[-1][1]
        c.execute("DELETE FROM scores WHERE (name = ? AND score = ?)",
                  (lowScoreName, lowScore))
    c.execute("INSERT INTO scores VALUES (?, ?, ?)", entry)
    conn.commit()
    conn.close()
```

shooting_game.py

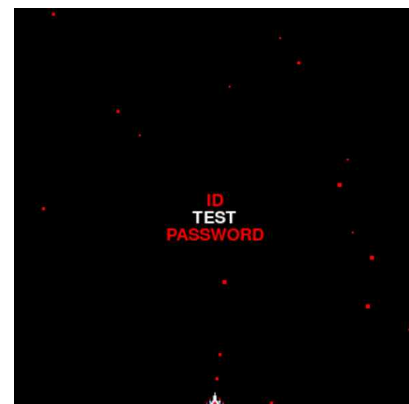
```
elif (event.type == pygame.KEYDOWN
      and event.key == pygame.K_RETURN
      and len(name) > 0):
    Database.setScore(hiScores, (name, score, accuracy))
    return True
```

- 서버를 이용해 계정을 생성하고, 로그인과 로그아웃 기능을 추가하여 플레이 정보를 서버와 주고받을 수 있게 함
- 계정을 생성하고 플레이기록을 서버에 저장하여 다른 사용자와 점수를 경쟁할 수 있음

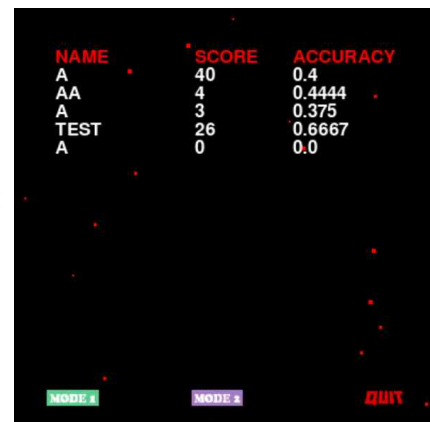
- 로그인 전 화면




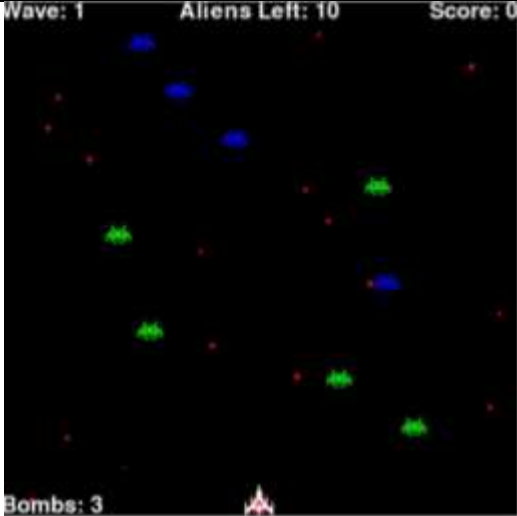


- 계정 생성 화면



- 다른 사용자와 점수 경쟁



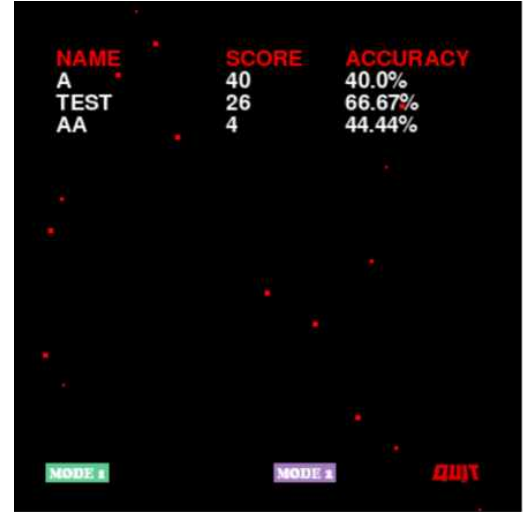
<p>마우스로 버튼을 클릭하여 모드 변경</p>	<p>- 마우스를 사용하여 메뉴를 선택할 수 없음</p>	<p>- 메뉴화면에 마우스로 선택할 수 있는 모드선택 버튼과 quit버튼 생성</p>  <p>- 버튼 구현 클래스 작성</p> <pre>class Button: def __init__(self, gameDisplay, img_in, x, y, width, height, img_act, x_act, y_act, action): self.lvl_size = 0 mouse = pygame.mouse.get_pressed() click = pygame.mouse.get_pressed() if x + width > mouse[0] > x and y + height > mouse[1] > y: gameDisplay.blit(img_act, (x_act, y_act)) if click[0] and action == 'quitgame': pygame.quit() sys.exit() elif click[0] and action == 'mode_one': self.lvl_size = -1 elif click[0] and action == 'mode_two': self.lvl_size = -2 else: gameDisplay.blit(img_in, (x, y)) def draw(self, screen): # Blit the text. screen.blit(self.txt_surface, (self.rect.x+5, self.rect.y+5)) # Blit the rect. pygame.draw.rect(screen, self.color, self.rect, 2)</pre> <p>- 메인 메뉴 아래에 버튼을 생성하는 코드</p> <pre>#버튼 구현 button1Pos = 0.08 #mode1 button2Pos = 0.52 #mode2 button3Pos = 0.82 #quit modeButton_one = Button(screen, modeImg_one, round(scr_size*button1Pos), round(scr_size* modeButton_two = Button(screen, modeImg_two, round(scr_size*button2Pos), round(scr_size* quitButton = Button(screen, quitImg, round(scr_size*button3Pos), round(scr_size*0.9), ro if modeButton_one.lvl_size == -1 : return scr_size, 1, id if modeButton_two.lvl_size == -2 : return scr_size, 1.6, id pygame.display.flip() #여기까지 버튼 구현</pre>
<p>Life 기능 추가</p>	<p>- 적과 한번 충돌 시 바로 게임 오버</p>	<p>- life기능을 추가하여 적과 충돌할 때 마다 오른쪽 위의 비행기 아이콘이 하나씩 사라짐, 비행기 아이콘이 다 사라지면 게임 종료됨.</p>

		
서버 기능	<ul style="list-style-type: none"> - 서버를 사용하지 않음 	<ul style="list-style-type: none"> - ubuntu에 pem키를 400권한으로 저장하여, root 권한으로 EC2 접속 - api만을 이용한 단순한 기능 구현, 파이썬 웹프레임워크인 fastapi 이용 - fastapi 공식문서를 참고하여, python3.8 이미지에 fastapi, uvicorn을 빌드하는 Dockerfile을 만들고, docker-compose.yml 파일을 작성하여 컨테이너를 빌드함
4주차		
인터페이스	<ul style="list-style-type: none"> - 게임을 영어로만 플레이 할 수 있음 <pre> startText = font.render('START GAME', 1, BLUE) startPos = startText.get_rect(midtop=titleRect.inflate(0, 100).midbottom) hiScoreText = font.render('HIGH SCORES', 1, BLUE) hiScorePos = hiScoreText.get_rect(topleft=startPos.bottomleft) fxText = font.render('SOUND FX ', 1, BLUE) fxPos = fxText.get_rect(topleft=hiScorePos.bottomleft) fxOnText = font.render('ON', 1, RED) fxOffText = font.render('OFF', 1, RED) fxOnPos = fxOnText.get_rect(topleft=fxPos.topright) fxOffPos = fxOffText.get_rect(topleft=fxPos.topright) musicText = font.render('MUSIC', 1, BLUE) musicPos = fxText.get_rect(topleft=fxPos.bottomleft) musicOnText = font.render('ON', 1, RED) musicOffText = font.render('OFF', 1, RED) musicOnPos = musicOnText.get_rect(topleft=musicPos.topright) musicOffPos = musicOffText.get_rect(topleft=musicPos.topright) quitText = font.render('QUIT', 1, BLUE) quitPos = quitText.get_rect(topleft=musicPos.bottomleft) selectText = font.render('*', 1, BLUE) selectPos = selectText.get_rect(topright=startPos.topleft) </pre>	<ul style="list-style-type: none"> - 언어(LANGUAGE)를 선택하면 영어와 한글 인터페이스가 반전됨, 로그인 후 언어를 선택한 상태로 게임을 종료하면 언어 정보가 계정에 저장되어 이후 게임 접속 시 반영됨 

- 중복된 아이디도 하이스코어 랭킹에 표시됨
- 정확도(Accuracy)가 소수 형태로 표기



- 정확도(Accuracy)표기를 퍼센테이지(%)로 변경하고, 중복되는 아이디는 최고 점수만 표시되도록 함



코인 시스템

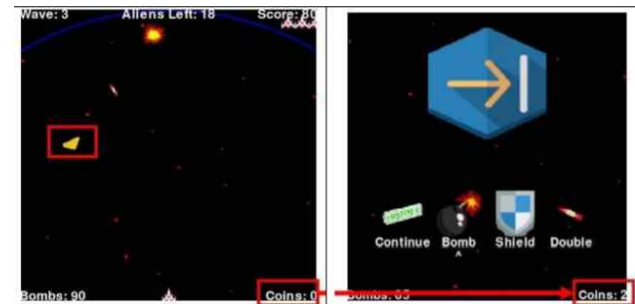
- 아이템은 게임 중에 드랍되는 것만 사용할 수 있으며, 별도로 구매할 수 있는 시스템이 없음

- 아이템과 같이 랜덤으로 드랍되는 코인을 추가하여 코인을 모아 shop에서 아이템을 구매할 수 있도록 함

- 코인 이미지



- 코인 드랍과 코인 획득





- 스테이지 별로 적들이 10, 20, 40으로 증가하며 등장하고, 해당 스테이지 적들을 모두 처치했을 때 다음 wave로 넘어가기 전에 3초의 간격이 있음

- Wave 이동 구간에서 카운트다운을 5초로 늘리고, 5초내에 키보드 i버튼을 누르면 shop으로 이동
- 현재 보유하고 있는 코인이 있다면 아이템 구매 가능(아이템 가격은 1코인으로 동일)

- Bombs는 1이고 coins는 2인 상황에서 Bomb을 2개 구매했을 때 bombs는 3 coins는 0으로 감소

한 것을 확인할 수 있음



<p>업적</p>	<ul style="list-style-type: none"> - 하이스코어(이름, 점수, 정확도)를 제외한 플레이 정보가 저장되지 않음 	<ul style="list-style-type: none"> - 미사일을 발사한 횟수(shoot)와 적을 무찌른 횟수를 저장하여 로그인 후에 업적(ACHIEVEMENTS)에서 확인 가능하도록 함 - 10, 100, 1000회 마다 배지가 추가됨 - Kill 10회  <ul style="list-style-type: none"> - Shoot 151회, kill 41회 
<p>서버 기능 추가 비동기 예외처리</p>	<ul style="list-style-type: none"> - 로그인, 하이스코어 관련 기능만 구현됨 - 인터넷 연결이 안 되는 경우 게임이 종료됨 	<ul style="list-style-type: none"> - 업적 관련 기능 추가 구현 - 비동기를 지원하는 grequests 라이브러리를 이용하여, 요청이 느린 경우 대기함 - 요청에 실패한 경우, 로그아웃된 메뉴 화면을 반환

5주차		
<div> <div>하이스코어 창에서</div> <div>아이디별 업적</div> <div>아이콘 추가</div> </div>	<div> <div>- 업적은 개인 업적 창에서 혼자만 볼 수 있었음</div> </div>	<div> <div>- hiscore 창에서도 개인의 업적을 볼 수 있는 아이콘 생성</div> <div> <div>이름</div> <div>점수</div> <div>정확도</div> <div>A</div> <div>40</div> <div>40.0%</div> <div>TEST</div> <div>35</div> <div>23.18%</div> <div>AA</div> <div>5</div> <div>83.33%</div> </div> </div>
<div> <div>한글 미구현 부분</div> <div>추가 구현</div> </div>	<div> <div>- 메인메뉴를 제외한 부분이 한글화 되지 않음</div> </div>	<div> <div>- 로그인, 업적, 하이스코어, 인게임에서 한글화 되도록 수정</div> <div> <div>웨이브: 2</div> <div>적 남은 수: 0</div> <div>점수: 20</div> <div>3 웨이브 남은 시간</div> <div>아이템 상점 : 1를 누르세요</div> </div> </div>
<div> <div>전체 코드 정리</div> </div>	<div> <div>- 반복되는 코드, 정리되지 않은 변수, 기타 버그 다수 존재</div> </div>	<div> <div>- 1인 모드를 기반으로 2인 모드를 개발하기 전에, 기존 모든 코드를 정리함</div> <div>- Resize로 변하는 모든 값을 size 클래스 변수로 모아서 선언</div> <div>- 반복되는 코드들을 함수로 구현</div> </div>
<div> <div>기타 버그</div> <div>수정 내용</div> </div>	<div> <div>- 언어 변환 시, 텍스트 길이가 달라서 텍스트가 겹치는 버그가 존재</div> </div>	<div> <div>- 버그 수정, 버튼 위치 수정</div> <div>- 기타 메뉴 이동이 안되는 페이지, 아이템 오류 수정</div> </div>

Wave: 4 Aliens Left: 0 Score: 339



Wave: 4 Aliens Left: 0 Score: 72

