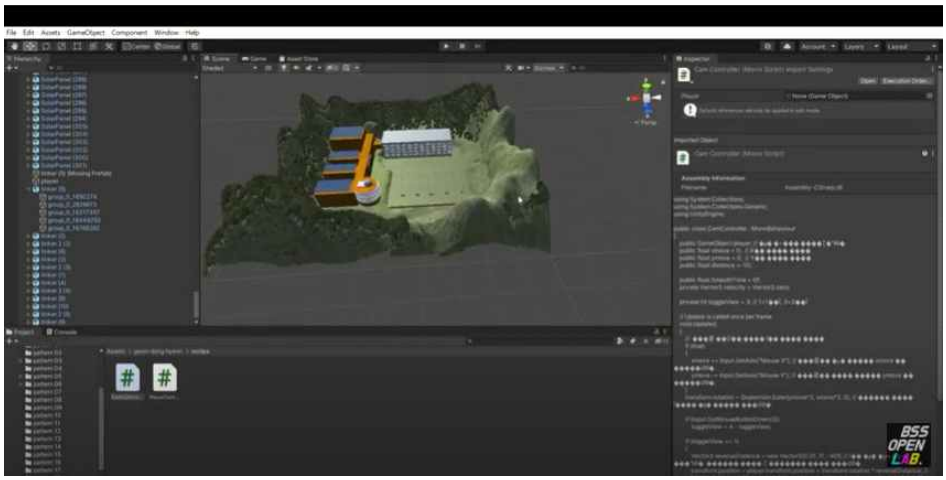
UNITY

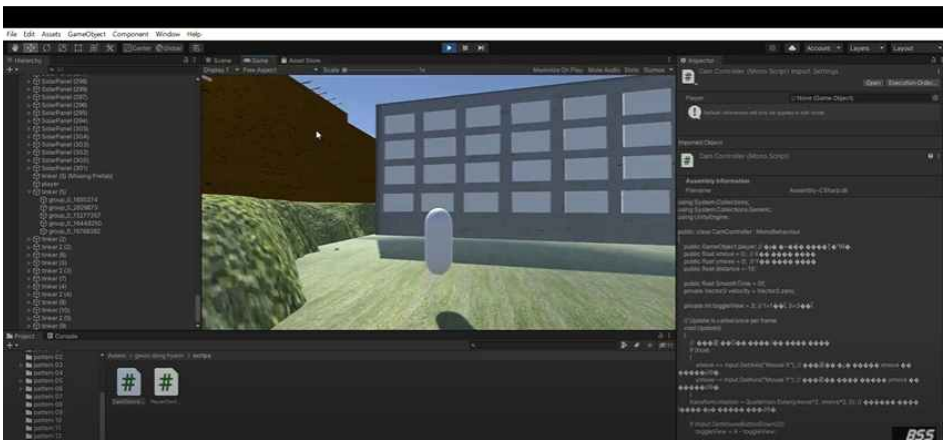
슈슉...슈슉...슉 레이저 양궁으로 금메달 도전하기(BIT)에서 unity로 양궁 게임을 만드는 것을 담당하였다. 이 활동을 통해서 저는 유니티를 이용하여서 3d게임을 만들어보면서 플레이어의 이동과 카메라의 이동에 대한 코드를 작성해 보았고 이를 통해서 unity와 c#을 공부하면서 저의 꿈인 프로그래머에 대한 역량을 늘렸습니다. 플레이어의 이동은 점프, 4방향 이동에 관련된 코드를 작성하였다.

〈카메라의 이동에 관해 작성한 코드일부〉

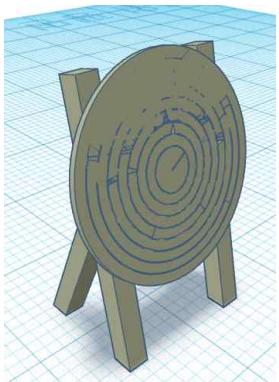
또한 thinkercad를 이용하여서 기숙사를 모델링하고 asset을 이용하여서 부산과학고 모양의 맵을 제작하는데 일조하였다.



〈제작에 참여한 부산과학고 맵의 모습〉



〈프로그래밍한 코드를 바탕으로 플레이어가 이동하는 모습〉



〈hinker cad에서 모델링한 양궁 과녁〉

또한 양궁게임에서 이스터에그를 넣는등 게임의 전반적인 게임의 기획등을 담당하였다.

Ren’Py

Ren’Py란?

Ren’Py 비주얼 노벨 엔진으로 오픈 소스 게임 엔진이다. 렌'파이의 뜻은 일본어로 '연애'를 의미하는 렌아이(일본어: 恋愛)와 파이썬을 합친 것이다. 렌'파이는 프로그래밍 언어 파이썬과 파이게임(PyGame)을 기반으로 만들어졌고, 윈도우와 최신 버전의 맥 OS X, 리눅스 모두 지원한다. 뿐만 아니라 손쉽게 안드로이드 플랫폼의 빌드가 가능하다.

첫 번째, 스토리 구성

민연시는 미소녀 연애 시뮬레이션 게임의 줄임말에서 유래한 신조어인 미연시에서 따온 말로 부산과학고 2학년 재학생이자 개발 총괄자 공민재를 대상으로 하여 연애를 하는 내용의 시뮬레이션으로 기획, 제작된 게임이다. 민연시는 ‘썸썸 편의점’, 기적의 분식점‘ 등의 시중에 판매되고 있는 미연시 게임들의 스토리 구성을 참고하여 제작되었다. 플레이어는 게임 속 여주인공이 되어 남주 ’민재‘와 여러 사건 사고를 겪으며 많은 선택지를 마주하게 되고 이때 선택한 여러 요소들을 통해 민재와의 호감도, 근력, 지력, 매력, 인성 수치를 올려가며 스토리를 진행한다. 이후 높은 수치에 따라 서울대 엔딩, 헬창 엔딩, 귀신 엔딩, 연애 성공 엔딩 등을 보게 된다.

두 번째, 인 게임 디자인

민연시는 전체적으로 대표적인 미연시 게임인 썸썸 편의점을 기반으로 제작된 게임으로 썸썸 편의점의 메인 메뉴, UI(User Interface, 사용자 인터페이스) 등의 전체적인 디자인을 중심으로 하여 인 게임 디자인을 제작하였다.

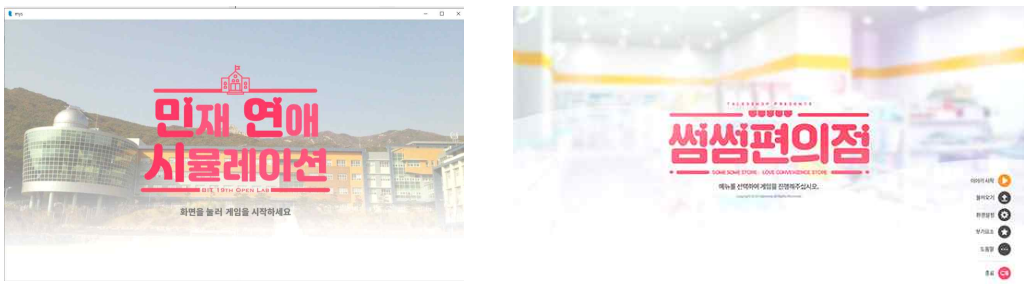


그림 1. 민연시와 썸썸 편의점의 메인 화면

게임을 실행 시 처음으로 나오는 메인 화면은 썸썸 편의점의 메인 화면을 참고하여 제작하였고, 이후 게임의 스토리 진행에는 우리 학교의 행사, 복도, 교실의 사진들을 촬영하여 제작하였다.

그림 2. 게임 플레이 화면(1)

그림 3. 게임 플레이 화면(1)

각각의 스토리가 진행됨에 따라 스토리의 진행과 관련된 배경 사진을 촬영하거나 검색하여 사진을 수집 후 적절한 위치에 삽입하였고 또 게임의 전체 진행이 게임의 남주인공 '민재'를 중심으로 진행되므로 여러 자세와 표정의 사진을 촬영 후 이를 적절한 위치에 삽입하여 게임의 몰입도를 높였다.

세 번째, 프로그래밍

1. script.rpy 작성 (대사, 이미지를 다루는 파일)

```
image bg school = 'images/bhs.jpg'
image bg room = 'ui/room.png'
image bg stair = 'bg/계단 사진.jpg'
image bg money = 'bg/땅에 떨어진 돈.jpg'
image bg hospital = 'bg/병원.jpg'
image bg bss_test = 'bg/부과고 면접.jpg'
image bg corridor = 'bg/부과고 복도.jpg'
```

그림4. 게임에 들어갈 배경을 정의

```
image more_disgust = '더역겹.png'
image angry = '소노.png'
image lovely_m = '애교민재.png'
image happy_m = '웃는민재.png'
image nomal_m:
    '평범민재.png'
    yalign 0
```

그림 5. 게임에 들어갈 인물 이미지를 정의

위의 두 사진과 같이 배경, 등장인물의 이미지들의 경로를 먼저 지정해주었다.

```
define m = Character('민재', color="#444444", what_color="#00ab60", what_slow_cps = 18)
define s = Character("[u_name]", color="#444444", what_color="#456788", what_slow_cps = 18)
define nar = Character(None, color="#444444", what_color="#666666", kind=None)
define hs = Character("홍성", color="#444444", what_color="#666666", kind=None)
define rdk = Character("다경", color="#444444", what_color="#666666", kind=None)
```

그림 6. 게임에 등장하는 인물의 약자 정의

그리고 위와 같이 게임에 등장하는 인물들의 이름 약자를 지정해주었다. 이름의 색, 대사의 색, 말하는 속도, 이름창에 들어갈 이름 등을 정의해준다. '[u_name]'에서 대괄호는 변수임을 알려주는 기호이고, u_name은 플레이어의 이름을 저장하는 변수이다.

```
init -3:
    $ u_name = ""
    $ u_int = 0
    $ u_man = 0
    $ u_str = 0
    $ u_att = 0
    $ u_tir = 10
    $ progress_num = [0,0, 3, 8, 20, 37, 52, 60, 70, 75, 100]
    $ date = 30
    $ progress = 1
    $ m_love = 0
    $ chocheck = 0
    $ exam = 0
    $ randomnum = 0
    $ score=0
```

그림 7. 게임에서 사용하는 변수명 정의

그리고 init 문을 사용해 게임에서 사용되는 변수명을 정의했다. renpy특성상 변수는 init이라는 구문을 사용해 정의해주어야한다. renpy는 모든 코드가 동시에 시행되게 되어있는데 init은 그중에서도 제일 먼저 시행하라는 의미를 담고있는 구문이다. 플레이어의 이름, 지력, 인성, 근력, 매력, 피로, 게임의 진행도, 시험 성적 등 다양한 변수를 정의했다.

```
label start:
    scene bg school
    with dissolve
    show screen list_button
    $ u_name = renpy.input("{color=#000000}당신의 이름을 입력해 주세요(성 제외) {color=#888888}(이름을 입력하지 않으면 '소희'라고 설정됩니다.)")
    $ u_name = u_name.strip()
    if u_name == "" :
        $ u_name = "소희"
    jump l1_1
    return
```

게임의 첫 시작 label은 이러하다. 먼저 플레이어에게 이름을 묻는다. renpy.input()을 사용해 이름을 입력받게 했고, 이름을 입력하지 않는 경우는 '소희'라는 default값으로 지정되도록 했다. 이름을 입력하면 jump명령어로 l1_1레이블로 화면이 넘어간다.

```
label lobby:
    play music "audio/bgm/room.mp3"
    show screen list_button
    scene bg room
    with dissolve
    while True:
        call screen lobby()
```

lobby화면을 불러오는 label이다. call screen 명령어로 lobby라는 스크린을 불러오도록 했고, call은 show와는 다르게 screen을 보여준다음 사용자의 입력을 기다리는 특징이 있기 때문에 show 대신 call을 사용했고, 사용자의 입력후 화면이 사라지는 것을 방지하기 위해 while문으로 무한반복 시켰다.

```
label l2_1:
    play music "audio/bgm/all.mp3"
    scene bg corridor2
    with dissolve
    s "뭔 놈의 학교가 프린트를 이렇게 많이 줘; 으악!"
    s "헉 미안! 안 다쳤어?"
    show lovely_m
    nar "프린트를 주워주는 민재와 손이 스쳤다"
    #뚜 뚜뚜 뚜 그 사랑 노래 넣기
    $ progress += 1
    jump lobby
```

여러개의 스토리 label중 하나를 가져왔다. play music 명령어로 음악을 재생했다. 대사가 출력되도록 하려면 대사 앞에 인물 이름을 적어야 한다는 점을 알 수 있을 것이다. 변수의 값을 바꾸려면 progress라는 변수 앞에 $을 넣어야 한다. 스토리가 끝나고 나면 jump 명령어로 lobby label로 넘어가도록 게임이 진행되도록 했다.

2. screens.rpy 작성(게임의 전체 인터페이스)

```
################################################################
## 게임내 스크린
################################################################
screen list_button:
    imagebutton auto "button/list_button_%s.png":
        xalign 0.98
        yalign 0.02
        action Show("in_list_button")

screen fast_button:
    imagebutton auto "button/fast_button_%s.png":
        xalign 0.98
        yalign 0.1
        action Skip()

screen in_list_button:
    frame:
        xsize 150
        ysize 720
```

그림 8 게임 내 버튼 설정

screens.rpy에서는 게임 내에 들어가는 다양한 버튼을 정의해놓는다. 스토리 스킵버튼, 환경설정 버튼 등등의 버튼이다.

그림 9 게임에서 보여지는 화면구성 (예시로 스토리가 끝나면 돌아가게 되어있는 lobby화면의 스크린 코드가 있다.)

위와 같이 screens.rpy에서는 게임에서 보여지는 다양한 화면들의 구성을 정의한다. lobby화면, 경고 화면, 안내 화면 등등이 있다.

screens.rpy에는 이 외에도 기본적으로 작성된 많은 화면과 버튼 코드들이 있지만 기존의 default로 지정된 코드에서 거의 바꾼 것이 없어서 소개하지 않겠다.

3. options.rpy

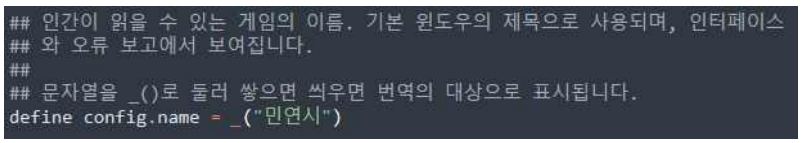


그림 10 윈도우 창에서의 게임 이름 정의

그림 11 배포판 (앱) 에서의 게임 이름 정의

이 파일에서는 게임 외적인 부분에 대한 것들을 작성하는 파일이다. 게임의 이름, 버전, 아이콘 등등의 요소들을 정의한다.

4. gui.rpy

이 파일에는 게임 내 UI와 관련된 코드가 작성되어있다.

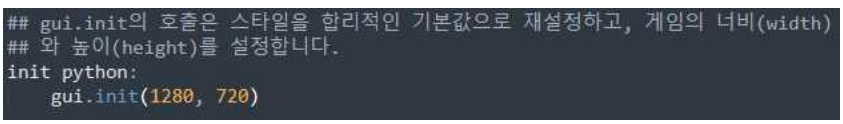


그림 12 pc에서 게임의 창 크기 설정

```
## 색상 ############################################################
##
## 인터페이스에서 글자의 색상입니다.

## 강조 색상은 레이블(label)과 강조된 글자로 인터페이스 전체에서 사용됩니다.
define gui.accent_color = u'#cc0066'
```

그림 13 게임 내 글자 색상 설정

```
## 글자와 글자 크기 ###################################

## 인-게임 글자에 사용됩니다.
define gui.text_font = "NanumBarunGothicBold.ttf"

## 캐릭터의 이름에 사용됩니다.
define gui.name_text_font = "NanumBarunGothicBold.ttf"
```

그림 14 게임 내 글자 폰트 지정

게임의 전체적인 글자 폰트는 나눔바른고딕 체로 하였다.

```
## 일반 대사의 글자 크기입니다.
define gui.text_size = 26
## 캐릭터 이름의 글자 크기입니다.
define gui.name_text_size = 25

## 게임의 유저 인터페이스에서 글자의 크기입니다.
define gui.interface_text_size = 22
```

그림 15 게임 내 글자 크기 지정

```
## 대사 ############################################################################
##
## 이러한 변수들은 한 번에 한 줄의 대사가 어떻게 화면에 표시되는지 제어합니다.

## 대사를 포함하는 텍스트 박스의 높이입니다.
define gui.textbox_height = 210

## 화면에 텍스트박스를 세로로 배치합니다. 0.0은 최상단, 0.5는 중앙, 그리고 1.0은
## 최하단입니다.
define gui.textbox_yalign = 0.99
```

그림 16 게임 내 대화창 크기, 위치 설정

```
########################################################################################
## 모바일 기기
########################################################################################

init python:

    ## 이것은 휴대전화와 태블릿에서 쉽게 터치할 수 있도록 빠른(Quick) 버튼들의
    ## 크기를 크게 합니다.
    if renpy.variant("touch"):

        gui.quick_button_borders = Borders(40, 14, 40, 0)

    ## 이것은 휴대전화에서 다양한 GUI 요소들의 크기와 간격을 쉽게 보일 수 있도록
    ## 변경합니다.
    if renpy.variant("small"):
```

그림 17 모바일기기에서의 추가 UI설정