Tetris

2016003609 성진아

구현한 기능들

기본 기능

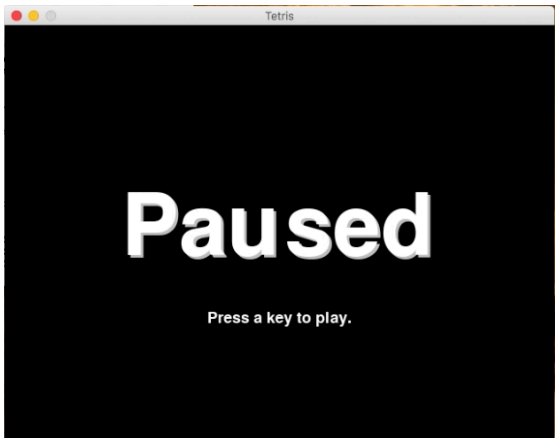
- 7 종류의 블럭이 랜덤하게 내려온다.
- 특정 키(Q / E / ↑)를 눌러 블럭을 회전시킬 수 있다.
- 회전한 블럭이 아래에 쌓인다.
- 한 줄 가득 블럭이 쌓이면 그 줄이 지워진다.
- 특정 키를 눌러 블럭을 좌우로 움직인다. (좌 : A or ← // 우 : D or →)
- 특정 키(S / ↓)를 눌러 블럭을 아래로 한 칸 움직인다.

추가 기능

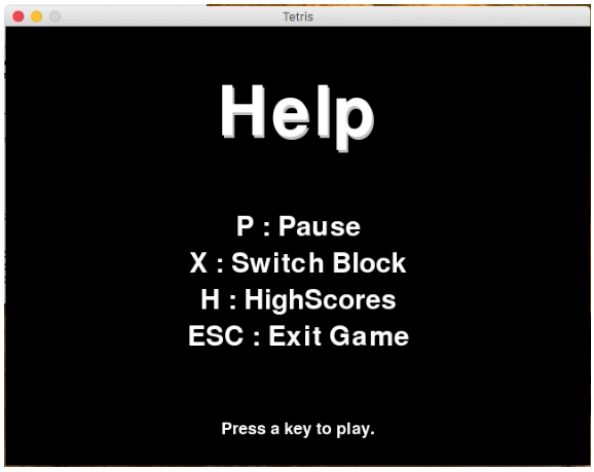
- 시작할 때 'tetris'라는 창이 뜨고, 아무 키나 입력하면 시작한다.

- 한 줄 지울 때마다 score 가 올라간다.

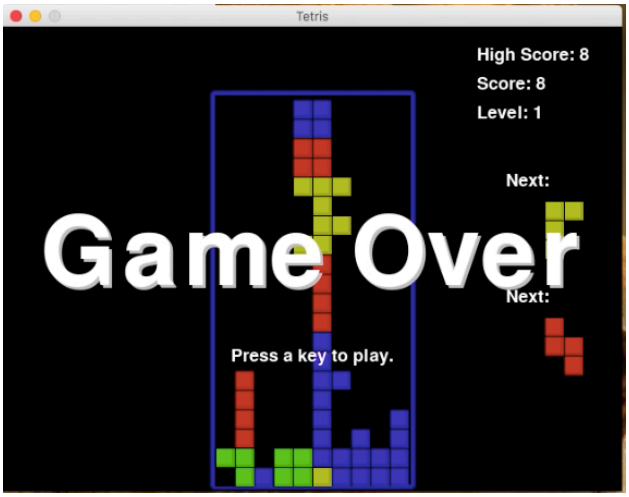
- 최고 score 에 도달할 때마다 highscore 가 갱신된다.
  ⇒ score.txt 파일에 갱신한 highscore 를 저장하고, 새 게임을 시작할 때 파일을 불러오는 식으로 구현함
  ⇒ 그렇기 때문에 score.txt 파일이 없으면 실행되지 않을 수 있다!
- score 가 10 씩 올라갈 때마다 level 이 1 씩 올라간다. (처음 level 은 1 이다.)
- level 이 올라갈 때마다 블럭이 바닥에 도달하는 속도를 줄인다. (속도를 빠르게 한다.)
  ⇒ 원래 속도에서 level * 0.2 만큼 빼는 형식으로 구현.
- 화면에 현재까지 제일 높은 점수(high Score)를 보여준다.
- 다음 블럭과 다다음 블럭을 보여준다.
- 특정 키(X)를 눌러 떨어지고 있는 블럭을 다음에 나올 블럭과 교체할 수 있다.
  ⇒ 키가 눌리면 떨어지고 있는 블럭의 위치에 next 블럭의 모양을 갖고 오고,
  ⇒ 떨어지고 있던 블럭의 위치를 초기화 시킨 후 next 블럭으로 바꾸는 식으로 구현함
- 배경음 2 가지를 새롭게 게임을 시작할 때 랜덤하게 틀어준다.
  ⇒ tetrisb.mid / tetrisc.mid
- 특정 키(ESC)를 눌러 게임을 종료한다.
- 특정 키(P)를 눌러 게임을 잠깐 멈춘다.



- 특정 키(O)를 눌러 도움 화면을 보여준다.
  ⇒ 어떤 키가 어떤 역할을 하는지에 대해서!



- 특정 키(H)를 눌러 역대 최고 점수들을 보여준다.
  ⇒ 게임이 끝났을 때, 최고점수가 갱신되었다면 high.txt 파일의 제일 밑줄에 점수를 써넣는 식으로 구현함 ⇒ highScore 들의 log 라고 보면 된다.
- 특정 키(Space)를 눌러 블럭이 바닥으로 바로 떨어지게 한다.
- 더 이상 블럭이 내려올 수 없을 때(블럭이 맨 위에 닿았을 때), 'Game Over' 창이 뜬다. 이 때, 아무 키나 입력하면 재시작한다.



(주의사항! 키보드가 한글로 되어있을 때, 일부 키가 안 먹힐 수 있다.)

추가한 알고리즘 설명

- 특정 키(H)를 눌러 역대 최고 점수들을 보여준다.

```
elif (event.key == K_h):
    DISPLAYSURF.fill(BGCOLOR)
    pygame.mixer.music.stop()
    showHigh() # pause until a key press
    pygame.mixer.music.play(-1, 0.0)
    lastFallTime = time.time()
    lastMoveDownTime = time.time()
    lastMoveSidewaysTime = time.time()
```

H 키가 눌리는 순간에 음악이 멈추고 showHigh() 함수를 실행시킨 후에 음악이 다시 재생된다.

```
initialHigh = getHighScore()
runGame(initialHigh)
highSave = getHighScore()
if initialHigh != highSave :
    highSaveFile = open("high.txt",'a')
    highSaveFile.write(str(highSave)+"\n")
    highSaveFile.close()
showTextScreen('Game Over')
```

high.txt 파일에 새로운 highScore 가 생길 때마다 맨 밑줄에 이어쓰도록 만들었다.

```
    topList = [0,0,0]
    num = 0
    high = open("high.txt", 'r')
    while True:
        line = high.readline()
        if not line: break
        if num > 2 :
            topList.append(0)
        topList[num] = int(line)
        num += 1
    high.close()
    length = len(topList)

    first = topList[length-1]
    second = topList[length-2]
    third = topList[length-3]

    if third != 0 and first == 0 and second == 0: # 파일에 한 줄만 있는 경우
        first = third
        third = 0
    if third != 0 and first == 0 and second != 0: # 파일에 두 줄만 있는 경우
        first = second
        second = third
```

```
        third = 0


    firstSurf, firstRect = makeTextObjs('1 :    ' + str(first), MYFONT,
TEXTCOLOR)
    firstRect.center = (int(WINDOWWIDTH / 2), 220)
    DISPLAYSURF.blit(firstSurf, firstRect)

    secondSurf, secondRect = makeTextObjs('2 :    ' + str(second), MYFONT,
TEXTCOLOR)
    secondRect.center = (int(WINDOWWIDTH / 2), 280)
    DISPLAYSURF.blit(secondSurf, secondRect)

    thirdSurf, thirdRect = makeTextObjs('3 :    ' + str(third), MYFONT,
TEXTCOLOR)
    thirdRect.center = (int(WINDOWWIDTH / 2), 340)
    DISPLAYSURF.blit(thirdSurf, thirdRect)

    pressKeySurf, pressKeyRect = makeTextObjs('Press a key to play.',
BASICFONT, TEXTCOLOR)
    pressKeyRect.center = (int(WINDOWWIDTH / 2), int(WINDOWHEIGHT / 2) + 200)
    DISPLAYSURF.blit(pressKeySurf, pressKeyRect)
```

high.txt 파일을 불러와서 젤 밑줄에 있는 것을 first, 그 윗줄에 있는 것을 second, 또 그 윗줄에 있는것을 third 로 지정해 순서대로 스크린에 보여준다

- 특정 키(X)를 눌러 떨어지고 있는 블럭을 다음에 나올 블럭과 교체할 수 있다.

```
elif event.key == K_x:
    savePiece = fallingPiece
    nextPiece['x'] = fallingPiece['x']
    nextPiece['y'] = fallingPiece['y']
    fallingPiece = nextPiece
    savePiece['x'] = int(BOARDWIDTH / 2) - int( TEMPLATEWIDTH / 2)
    savePiece['y'] = -2
    nextPiece = savePiece;
```

X 키가 눌리면 savePiece 에 현재 떨어지고 있는 fallingPiece 를 저장하고, nextPiece 에 현재 떨어지고 있는 block 의 위치를 저장한다.
그 후에 두 개를 바꿔주면 nextPiece 의 shape 를 갖고 있는 블럭이 현재 block 이 있던 위치로 온다.
떨어지고 있는 block 의 위치를 초기화 시킨 후 nextPiece 에 가져다놓는다.

- 화면에 현재까지 제일 높은 점수(high Score)를 보여준다.

```
def getHighScore():
    # Default high score
    highScore = 0
```

```
    highScoreFile = open("score.txt", "r")
    highScore = int(highScoreFile.read())
    highScoreFile.close()
    return highScore
while True: # game loop
    if random.randint(0, 1) == 0:
        sound = pygame.mixer.music.load('tetrisb.mid')
    else:
        sound = pygame.mixer.music.load('tetrisc.mid')
    pygame.mixer.music.play(-1, 0.0)
    initialHigh = getHighScore()
    runGame(initialHigh)
```

while True 문 일부를 가져온 것이다. 게임을 시작할 때 파일에 저장되어있는 highScore 를 불러온다.

```
def saveHighScore(newHighScore):
    highScoreFile = open("score.txt", "w")
    highScoreFile.write(str(newHighScore))
    highScoreFile.close()
if time.time() - lastFallTime > fallFreq:
    # see if the piece has landed
    if not isValidPosition(board, fallingPiece, adjY=1):
        # falling piece has landed, set it on the board
        addToBoard(board, fallingPiece)
        score += removeCompleteLines(board)

        if score > highScore :
            highScore = score
        level, fallFreq = calculateLevelAndFallFreq(score)
        fallingPiece = None

    else:
    # piece did not land, just move the piece down
        fallingPiece['y'] += 1
        lastFallTime = time.time()

saveHighScore(highScore)
```

highScore 보다 현재 Score 가 더 높다면, highScore 를 바꾸고 score.txt 파일에 덮어쓰기 해준다.