과제 중심 수업

ICT융합학부 2023054939 김은주

1. **현재 테트리스 게임의 배경음악을 주어진 3개의 음악 중 1개가 재생되도록 수정**

```
while True: # game loop
    if random.randint(0, 1) == 0:
        pygame.mixer.music.load('Hover.mp3')
    else:
        pygame.mixer.music.load('Hover.mp3')
    pygame.mixer.music.play(-1, 0.0)
    runGame()
    pygame.mixer.music.stop()
    showTextScreen('Game Over')
```

2개 모두 음악 파일이름으로 변경

2. **상태창 이름을 학번_이름으로 수정**

```
pygame.display.set_caption('2023054939_KIMEUNJU')
```

Main함수 아래에 학번_이름으로 변경

3. **게임시작화면의 문구를 MY TETRIS으로 변경**

```
showTextScreen('MY TETRIS')
```

Main함수 아래에 MY TETRIS로 변경

4. **게임시작화면의 문구 및 배경색을 노란색으로 변경**

```
titleSurf, titleRect = makeTextObjs(text, BIGFONT, YELLOW)

titleSurf, titleRect = makeTextObjs(text, BIGFONT, YELLOW)

pressKeySurf, pressKeyRect = makeTextObjs('Press a key to play.', BASICFONT, YELLOW)

scoreSurf = BASICFONT.render('Score: %s' % score, True, YELLOW)

levelSurf = BASICFONT.render('Level: %s' % level, True, YELLOW)

nextSurf = BASICFONT.render('Next:', True, YELLOW)

timeSurf = BASICFONT.render('Play Time: %s sec' % time, True, YELLOW)
```

TEXTCOLOR부분과 TEXTSHADOWCOLOR부분 모두 YELLOW로 변경

5. 게임 경과 시간을 초 단위로 표시 (새 게임 시작시 0으로 초기화 되어야 함)

```
def drawTime(time):
    timeSurf = BASICFONT.render('Play Time: %s sec' % time, True, YELLOW)
    timeRect = timeSurf.get_rect()
    timeRect.topleft = (WINDOWWIDTH -600, 20)
    DISPLAYSURF.blit(timeSurf, timeRect)

def runGame():
    # setup variables for the start of the game
    board = getBlankBoard()
    lastMoveDownTime = time.time()
    lastMoveSidewaysTime = time.time()
    lastFallTime = time.time()
    movingDown = False # note: there is no movingUp variable
    movingLeft = False
    movingRight = False
    score = 0
    level, fallFreq = calculateLevelAndFallFreq(score)

    fallingPiece = getNewPiece()
    nextPiece = getNewPiece()

    # Add this line to track the game time
    gameTime = 0

    # Add this line to update the timer display
    drawTime(gameTime)

    # Add a timer to update the game time
    lastTime = time.time()

    while True: # game loop
        # Add this block to update the game time
        currentTime = time.time()
        gameTime += currentTime - lastTime
        lastTime = currentTime

        # Add this line to update the timer display
        drawTime(int(gameTime))
```

초 단위로 경과 시간을 표시할 텍스트를 화면에 표시하는 drawTime 함수 새로 추가

runGame 함수 아래에 게임 경과 시간을 관리하고 표시하는 코드 추가

- gametime = 0
- drawTime(gameTime)
- lastTime = time.time()
- drawTime(int(gameTime))

6. 7개의 블록이 각각 고유의 색을 갖도록 코드를 수정하거나 추가

```
#               R    G    B
WHITE       = (255, 255, 255)
GRAY        = (185, 185, 185)
BLACK       = (  0,   0,   0)
RED         = (155,   0,   0)
LIGHTRED    = (175,  20,  20)
GREEN       = (  0, 155,   0)
LIGHTGREEN  = ( 20, 175,  20)
BLUE        = (  0,   0, 155)
LIGHTBLUE   = ( 20,  20, 175)
YELLOW      = (155, 155,   0)
LIGHTYELLOW = (175, 175,  20)
ORANGE = (255, 165, 0)
CYAN = (0, 255, 255)
PURPLE = (128, 0, 128)

BORDERCOLOR = BLUE
BGCOLOR = BLACK
TEXTCOLOR = WHITE
TEXTSHADOWCOLOR = GRAY
COLORS      = (     BLUE,      GREEN,      RED,      YELLOW, ORANGE, CYAN, PURPLE)

TEMPLATEWIDTH = 5
TEMPLATEHEIGHT = 5

COLORSS = {'S': BLUE, 'Z': GREEN, 'J': YELLOW, 'L': RED, 'I': PURPLE, 'O':
ORANGE, 'T': CYAN}

def getNewPiece():
    # return a random new piece in a random rotation and color
    shape = random.choice(list(PIECES.keys()))
    newPiece = {'shape': shape,
                'rotation': random.randint(0, len(PIECES[shape]) - 1),
                'x': int(BOARDWIDTH / 2) - int(TEMPLATEWIDTH / 2),
                'y': -2, # start it above the board (i.e. less than 0)
                'color': COLORSS[shape]}
    return newPiece

def drawBox(boxx, boxy, color, pixelx=None, pixely=None):
    # draw a single box (each tetromino piece has four boxes)
    # at xy coordinates on the board. Or, if pixelx & pixely
    # are specified, draw to the pixel coordinates stored in
    # pixelx & pixely (this is used for the "Next" piece).
    if color == BLANK:
        return
    if pixelx == None and pixely == None:
        pixelx, pixely = convertToPixelCoords(boxx, boxy)
    pygame.draw.rect(DISPLAYSURF, color, (pixelx + 1, pixely + 1, BOXSIZE - 1, BOXSIZE - 1))
    pygame.draw.rect(DISPLAYSURF, color, (pixelx + 1, pixely + 1, BOXSIZE - 4, BOXSIZE - 4))
```

색상 3개 추가, LIGHTCOLORS 삭제

- ORANGE = (255, 165, 0)

- CYAN = (0, 255, 255)

- PURPLR =(128, 0, 128)

PIECES 아래 색상 지정 딕셔너리 추가

COLORSS = {'S':BLUE, 'Z':GREEN, 'J':YELLOW, 'L':RED, 'I':PURPLE, 'O':ORANGE, 'T':CYAN}

getNewPiece 함수 아래에 random.randint(0, len(COLORS)-1)를 COLORSS[shape]으로 변경

drawBox 함수 아래에 COLORS[color], LIGHTCOLORS[color]를 color로 변경

각 함수의 역할

1 main

: Pygame 초기화, 디스플레이 및 글꼴 설정, 게임 루프 관리, 배경 음악 재생, 게임 시작 및 종료 처리 등 게임 실행을 위한 초기 설정과 메인 루프를 담당하는 핵심 함수이다.

-pygame.init()으로 pygame 초기화

-global FPSCLOCK, DISPLAYSURF, BASICFONT, BIGFONT으로 전역변수를 설정하여 다른 함수들에서도 접근할 수 있도록 함

-FPSCLOCK = pygame.time.Clock()으로 게임시계설정

-DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))으로 디스플레이 설정

-BASICFONT = pygame.font.Font('freesansbold.ttf', 18), BIGFONT = pygame.font.Font('freesansbold.ttf', 100)으로 글꼴 설정

-pygame.display.set_caption('Tetromino')으로 게임 창 제목 설정

-showTextScreen('Tetromino')으로 시작 화면 표시

-while True:으로 메인 게임 루프

-if random.randint(0, 1) == 0:으로 배경음악 로드 및 재생

-runGame()으로 게임실행

2 drawBox(boxx, boxy, color, pixelx=None, pixely=None):

함수는 주어진 좌표에 사각형(박스)을 그리는 역할을 한다. 테트리스 게임의 블록(테트로미노)의 각 구성 요소를 화면에 렌더링하는 데 사용되고 화면상의 위치를 계산하여 주어진 색상으로 사각형을 그린다.

-color가 blank이면 함수 바로 종료

-pixelc와 pixely가 주어지지 않으면 converToPixelCoords함수를 사용하며 boxx와 boxy를 픽셀단위 좌표로 변환

-pygame.draw.rect함수를 사용하여 두 개의 사각형을 그림

3 drawNextPiece(piece)

테트리스 게임 화면의 한쪽에 "다음 조각"을 미리 보여주는 역할을 한다.

-BASICFONT.render('Next:', True, TEXTCOLOR)는 "Next:"라는 텍스트를 렌더링하여 표면(surf) 객체를 생성함

-get_rect() 메서드는 이 표면의 사각형(rect) 객체를 가져옴

-nextRect.topleft를 통해 텍스트의 위치를 설정합니다. 여기서는 화면 오른쪽 상단에 "Next:" 텍스트를 표시함

-DISPLAYSURF.blit(nextSurf, nextRect)는 텍스트를 실제로 화면에 그림

-drawPiece 함수는 주어진 조각을 그림

-pixelx와 pixely 인수는 조각이 그려질 위치의 픽셀 좌표를 지정함

-WINDOWWIDTH - 120과 100 픽셀 위치에 다음 조각을 그림

함수의 호출 순서 및 호출 조건에 대한 설명

1 main() 함수

1)호출 조건: 프로그램 시작 시 직접 호출

2)호출 순서 및 조건:

-pygame.init(): Pygame 라이브러리 초기화

-pygame.time.Clock(): 프레임 레이트 제어를 위한 시계 객체 생성

-pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT)): 게임 창 설정

-pygame.font.Font('freesansbold.ttf', 18): 기본 폰트 설정

-pygame.font.Font('freesansbold.ttf', 100): 큰 폰트 설정

-pygame.display.set_caption('Tetromino'): 창 제목 설정

-showTextScreen('Tetromino'): 초기 화면에 "Tetromino" 텍스트 표시

-게임 루프 시작:

-랜덤한 음악 파일 로드 및 재생

-runGame() 함수 호출

-음악 중지 및 'Game Over' 텍스트 화면 표시

2 runGame() 함수

1)호출 조건: main() 함수의 게임 루프 내에서 호출

2)호출 순서 및 조건:

-getBlankBoard(): 빈 보드 생성

-초기 시간 설정: time.time()을 통해 초기 시간 기록

-초기 이동 상태 및 점수 설정

-calculateLevelAndFallFreq(score): 초기 레벨 및 낙하 주기 계산

-getNewPiece(): 새 테트리스 조각 생성

-getNewPiece(): 다음 테트리스 조각 생성

-게임 루프 시작:

-새로운 조각이 필요하면 새 조각 설정 및 유효성 검사

-사용자 입력 처리: 키 입력에 따라 조각 이동, 회전, 빠른 낙하 등 처리

-조각 이동 처리: 왼쪽, 오른쪽, 아래쪽 이동 및 자동 낙하 처리

-조각이 바닥에 닿으면 보드에 추가하고 점수 및 레벨 갱신

-화면 갱신: 보드, 상태, 다음 조각, 현재 조각 등을 그림

-프레임 업데이트: pygame.display.update() 및 FPSCLOCK.tick(FPS)

3 showTextScreen(text) 함수

1)호출 조건: 게임 시작 및 게임 오버 시 main() 함수에서 호출

2)호출 순서 및 조건:

텍스트 그림자 및 텍스트를 화면 중앙에 그림

"Press a key to play." 텍스트를 화면에 그림

키 입력 대기: checkForKeyPress()를 통해 키 입력이 있을 때까지 대기

4 checkForKeyPress() 함수

1)호출 조건: showTextScreen(text) 함수 내에서 호출.

2)호출 순서 및 조건:

-종료 이벤트 검사: checkForQuit() 호출

-키 입력 이벤트 검사: KEYDOWN 이벤트를 큐에서 제거하고 KEYUP 이벤트가 발생할 때까지 대기

-키 입력이 있으면 해당 키 반환, 없으면 None 반환

5 checkForQuit() 함수

1)호출 조건: 다양한 위치에서 게임 종료를 확인하기 위해 호출.

2)호출 순서 및 조건:

-QUIT 이벤트 검사: QUIT 이벤트가 발생하면 terminate() 호출

-ESC 키 검사: ESC 키가 눌리면 terminate() 호출

-나머지 KEYUP 이벤트는 다시 이벤트 큐에 추가

6 terminate() 함수

1)호출 조건: 게임을 종료할 때 호출

2)호출 순서 및 조건:

-pygame.quit(): Pygame 종료

-sys.exit(): 프로그램 종료

7 기타 그리기 함수 (drawBoard, drawStatus, drawPiece, drawNextPiece)

1)호출 조건: runGame() 함수의 게임 루프 내에서 호출

2)호출 순서 및 조건:

-drawBoard(board): 현재 보드를 화면에 그림

-drawStatus(score, level): 현재 점수와 레벨을 화면에 그림

-drawNextPiece(nextPiece): 다음 조각을 화면에 그림

-drawPiece(fallingPiece): 현재 떨어지고 있는 조각을 화면에 그림

8 조각 관련 함수 (getNewPiece, isValidPosition, addToBoard, removeCompleteLines)

1)호출 조건: 조각 생성, 이동, 보드 추가, 라인 제거 등 다양한 상황에서 호출

2)호출 순서 및 조건:

-getNewPiece(): 새 조각을 생성하여 반환

-isValidPosition(board, piece, adjX=0, adjY=0): 조각이 유효한 위치인지 검사

-addToBoard(board, piece): 조각을 보드에 추가

-removeCompleteLines(board): 완성된 라인을 제거하고 제거된 라인 수 반환

GitHub Respository 주소

https://github.com/HYUeunju/osw