

# Lecture #6. 입력 처리

2D 게임 프로그래밍

이대현 교수



한국공학대학교  
TECH UNIVERSITY OF KOREA

# 학습 내용

---

- 입력 처리 과정
- 키보드 입력 처리
- 마우스 입력 처리

# 키보드 및 마우스 입력 처리 과정

---

**Step1: 입력 이벤트들을 폴링한다.(get\_events())**



**Step2: 이벤트의 종류를 구분한다.(event.type 을 이용)**



**Step3: 실제 입력값을 구한다.(event.key 또는 event.x, event.y 등 을 이용)**



ESC 를 이용한 종료

# character\_runs\_esc.py (1)



```
running = True
```

```
def handle_events():
```

```
    global running
```

```
    events = get_events()
```

```
    for event in events:
```

```
        if event.type == SDL_QUIT:
```

```
            running = False
```

```
        elif event.type == SDL_KEYDOWN and event.key == SDLK_ESCAPE:
```

```
            running = False
```

# character\_runs\_esc.py (2)



```
frame = 0
for x in range(0, 800, 5):

    clear_canvas()
    grass.draw(400, 30)
    character.clip_draw(frame * 100, 100, 100, 100, x, 90)
    update_canvas()

    handle_events()
    if not running:
        break

    frame = (frame + 1) % 8
    delay(0.05)
```

# get\_events() - 발생한 모든 이벤트를 모아서 가져옴.

```
def handle_events():
```

```
    global running
```

이벤트들이 담긴 리스트가 넘어옴.

```
    events = get_events()
```

```
    for event in events:
```

이벤트를 하나씩 꺼내서 확인함.

```
        if event.type == SDL_QUIT:
```

```
            running = False
```

```
        elif event.type == SDL_KEYDOWN and event.key == SDLK_ESCAPE:
```

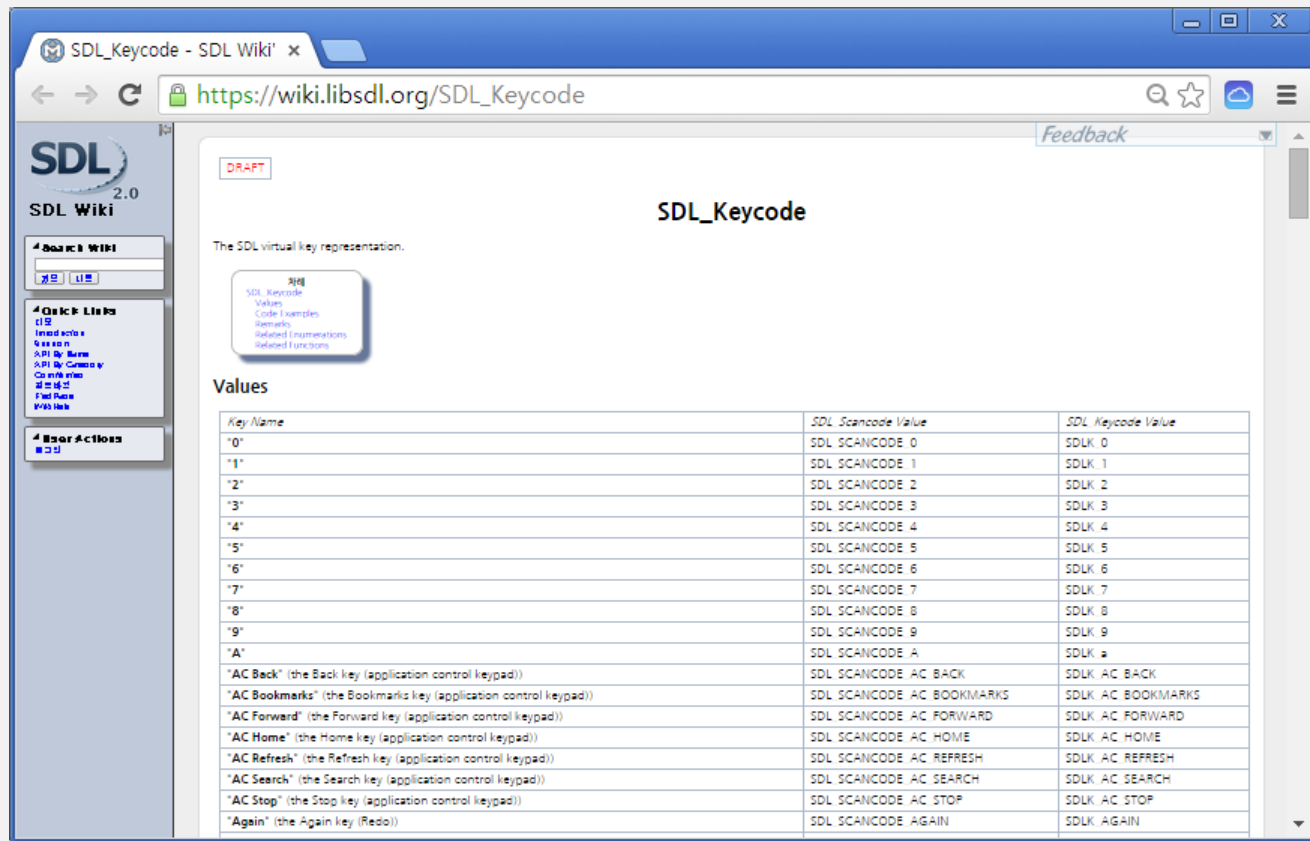
```
            running = False
```

## 이벤트 타입([https://wiki.libsdl.org/SDL\\_EventType](https://wiki.libsdl.org/SDL_EventType))

event.type	설명
SDL_QUIT	윈도우 종료 시 발생
SDL_KEYDOWN SDL_KEYUP	키가 눌리거나 떼어질 때 발생 event.key 에 key 값이 넘어옴
SDL_MOUSEMOTION	마우스가 움직일 때 발생 event.x, event.y 에 좌표값이 넘어옴
SDL_MOUSEBUTTONDOWN SDL_MOUSEBUTTONUP	마우스 버튼이 눌리거나 떼어질 때 발생 event.button 에 버튼의 종류(SDL_BUTTON_LEFT, SDL_BUTTON_MIDDLE, SDL_BUTTON_RIGHT)가, event.x, event.y 에 그 시점에서의 마우스 좌표값(기준점,왼 쪽위)이 넘어옴.
SDL_MOUSEWHEEL	마우스 휠이 움직일 때, 발생함. event.wheel.x, event.wheel.y 에 스크롤량이 넘어옴.



# SDL 키코드([https://wiki.libsdl.org/SDL\\_Keycode](https://wiki.libsdl.org/SDL_Keycode))



The screenshot shows a web browser window with the URL [https://wiki.libsdl.org/SDL\\_Keycode](https://wiki.libsdl.org/SDL_Keycode). The page title is "SDL\_Keycode". On the left, there is a sidebar with the SDL Wiki logo and navigation links. The main content area has a "DRAFT" label and a description: "The SDL virtual key representation." Below this is a small box labeled "차이" (Difference) comparing SDL Keycode Values, Code Examples, Remarks, Related Enumerations, and Related Functions. The "Values" section contains a table with three columns: Key Name, SDL Scancode Value, and SDL Keycode Value.

Key Name	SDL Scancode Value	SDL Keycode Value
"0"	SDL_SCANCODE_0	SDLK_0
"1"	SDL_SCANCODE_1	SDLK_1
"2"	SDL_SCANCODE_2	SDLK_2
"3"	SDL_SCANCODE_3	SDLK_3
"4"	SDL_SCANCODE_4	SDLK_4
"5"	SDL_SCANCODE_5	SDLK_5
"6"	SDL_SCANCODE_6	SDLK_6
"7"	SDL_SCANCODE_7	SDLK_7
"8"	SDL_SCANCODE_8	SDLK_8
"9"	SDL_SCANCODE_9	SDLK_9
"A"	SDL_SCANCODE_A	SDLK_a
"AC Back" (the Back key (application control keypad))	SDL_SCANCODE_AC_BACK	SDLK_AC_BACK
"AC Bookmarks" (the Bookmarks key (application control keypad))	SDL_SCANCODE_AC_BOOKMARKS	SDLK_AC_BOOKMARKS
"AC Forward" (the Forward key (application control keypad))	SDL_SCANCODE_AC_FORWARD	SDLK_AC_FORWARD
"AC Home" (the Home key (application control keypad))	SDL_SCANCODE_AC_HOME	SDLK_AC_HOME
"AC Refresh" (the Refresh key (application control keypad))	SDL_SCANCODE_AC_REFRESH	SDLK_AC_REFRESH
"AC Search" (the Search key (application control keypad))	SDL_SCANCODE_AC_SEARCH	SDLK_AC_SEARCH
"AC Stop" (the Stop key (application control keypad))	SDL_SCANCODE_AC_STOP	SDLK_AC_STOP
"Again" (the Again key (Redo))	SDL_SCANCODE_AGAIN	SDLK_AGAIN

# global 변수 지정

```
running = True

def handle_events():

    global running

    events = get_events()

    for event in events:
        if event.type == SDL_QUIT:
            running = False
        elif event.type == SDL_KEYDOWN and event.key == SDLK_ESCAPE:
            running = False
```

함수 내에서 값이 결정되는 변수는 지역 변수로 간주됨.  
따라서, 변수를 global 이라는 점을 알려려면 반드시 global 로 지정해야 함.

# running 변수를 이용한 루프 탈출

---

```
frame = 0
for x in range(0, 800, 5):

    clear_canvas()
    grass.draw(400, 30)
    character.clip_draw(frame * 100, 100, 100, 100, x, 90)
    update_canvas()

    handle_events()
    if not running:
        break

    frame = (frame + 1) % 8
    delay(0.05)
```



## 캐릭터의 좌우 이동

# move\_character\_with\_key.py



```
def handle_events():
    global running
    global x
    events = get_events()
    for event in events:
        if event.type == SDL_QUIT:
            running = False
        elif event.type == SDL_KEYDOWN:
            if event.key == SDLK_RIGHT:
                x += 10
            elif event.key == SDLK_LEFT:
                x -= 10
            elif event.key == SDLK_ESCAPE:
                running = False
```

```
running = True
x = 800 // 2
frame = 0

while running:
    clear_canvas()
    grass.draw(400, 30)
    character.clip_draw(frame*100, 100, 100, 100, x, 90)
    update_canvas()
    handle_events()
    frame = (frame + 1) % 8
    delay(0.05)
```

```
def handle_events():
    global running, x
    events = get_events()
    for event in events:
        if event.type == SDL_QUIT:
            running = False
        elif event.type == SDL_KEYDOWN:
            if event.key == SDLK_RIGHT:
                x = x + 10
            elif event.key == SDLK_LEFT:
                x = x - 10
            elif event.key == SDLK_ESCAPE:
                running = False

running = True
x = 800 // 2
```

좌우 키가 눌리면, x값을  
증가 또는 감소

# 좌우 이동 추가 구현 - 키이가 눌린 상태를 추적



```
def handle_events():
    global running, dir

    events = get_events()
    for event in events:
        if event.type == SDL_QUIT:
            running = False
        elif event.type == SDL_KEYDOWN:
            if event.key == SDLK_RIGHT:
                dir += 1
            elif event.key == SDLK_LEFT:
                dir -= 1
            elif event.key == SDLK_ESCAPE:
                running = False
        elif event.type == SDL_KEYUP:
            if event.key == SDLK_RIGHT:
                dir -= 1
            elif event.key == SDLK_LEFT:
                dir += 1
```



```
running = True
x = 800 // 2
frame = 0
dir = 0

while running:
    clear_canvas()
    grass.draw(400, 30)
    character.clip_draw(frame * 100, 0, 100, 100, x, 90)
    update_canvas()
    handle_events()
    frame = (frame + 1) % 8
    x += dir * 5
    delay(0.05)
```



변수 dir 을 이용하여, x 축상의 방향을 표시.





마우스를 이용한 캐릭터 이동

# move\_character\_with\_mouse.py (1)



```
TUK_WIDTH, TUK_HEIGHT = 1280, 1024
open_canvas(TUK_WIDTH, TUK_HEIGHT)
tuk_ground = load_image('TUK_GROUND.png')
character = load_image('animation_sheet.png')
```

## move\_character\_with\_mouse.py (2)



```
def handle_events():
    global running
    global x, y
    events = get_events()
    for event in events:
        if event.type == SDL_QUIT:
            running = False
        elif event.type == SDL_MOUSEMOTION:
            x, y = event.x, TUK_HEIGHT - 1 - event.y
        elif event.type == SDL_KEYDOWN and event.key == SDLK_ESCAPE:
            running = False
```

# move\_character\_with\_mouse.py (3)



```
running = True  
frame = 0  
x, y = TUK_WIDTH // 2, TUK_HEIGHT // 2  
hide_cursor()
```

# move\_character\_with\_mouse.py (4)



```
while running:  
    clear_canvas()
```

```
tuk_ground.draw(TUK_WIDTH // 2, TUK_HEIGHT // 2)  
character.clip_draw(frame * 100, 100 * 1, 100, 100, x, y)
```

```
update_canvas()  
handle_events()  
frame = (frame + 1) % 8  
delay(0.05)
```

# 마우스 좌표의 획득과 변환

---

```
if event.type == SDL_QUIT:
    running = False
elif event.type == SDL_MOUSEMOTION:
    x, y = event.x, TUK_HEIGHT - 1 - event.y
elif event.type == SDL_KEYDOWN and event.key == SDLK_ESCAPE:
    running = False
```

마우스가 이동하면, SDL\_MOUSEMOTION 이벤트가 발생  
event.x 및 y는, 윈도우 API 의 좌표계를 따름.  
pico2d 좌표계 변환 필요.