

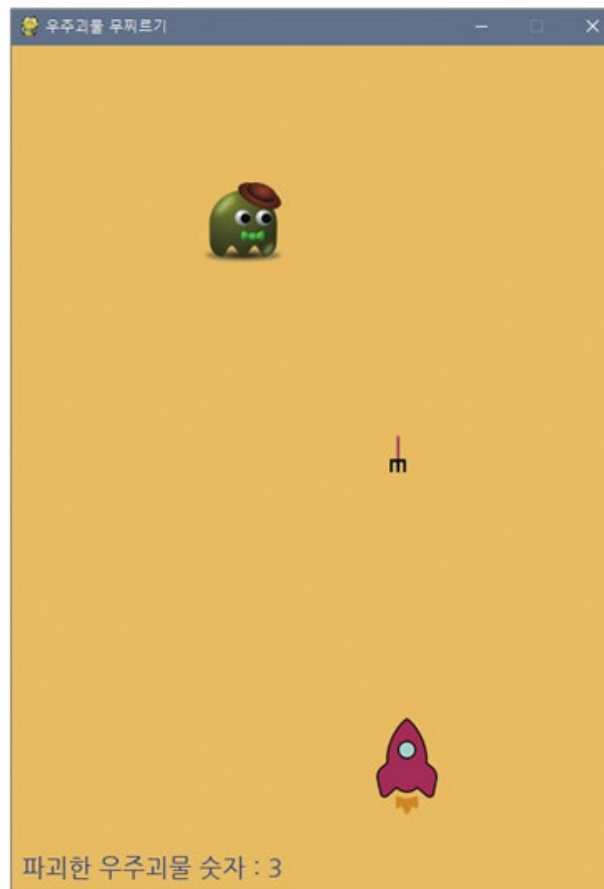
SECTION 02 슈팅 게임 프로젝트

요약

연습문제

■ 슈팅 게임 소개

■ 우주괴물 무찌르기



- 외부 라이브러리 설치

- pygame 외부 라이브러리 사용

```
pip install pygame
```

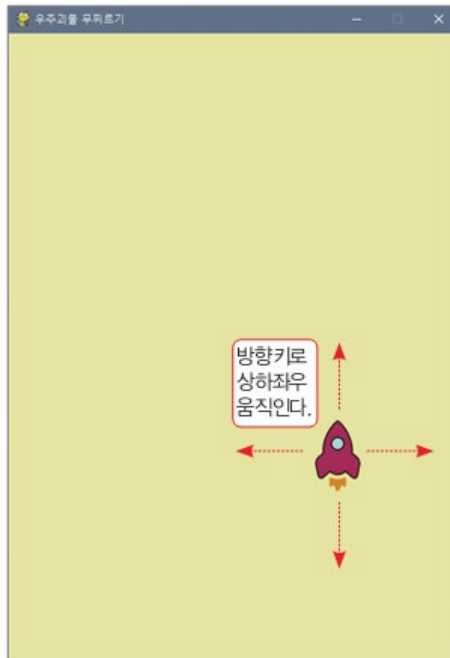
```
import pygame
```

■ 슈팅 게임 기능 구성

- 기능 1. 기본 화면 구성하기(code14-10.Py)



- 기능 2. 우주선 이미지를 추가하고 방향키로 움직이기(code14-11.Py)



- 기능 3. 우주괴물이 나타나면 자동으로 움직이기(code14-12.Py)



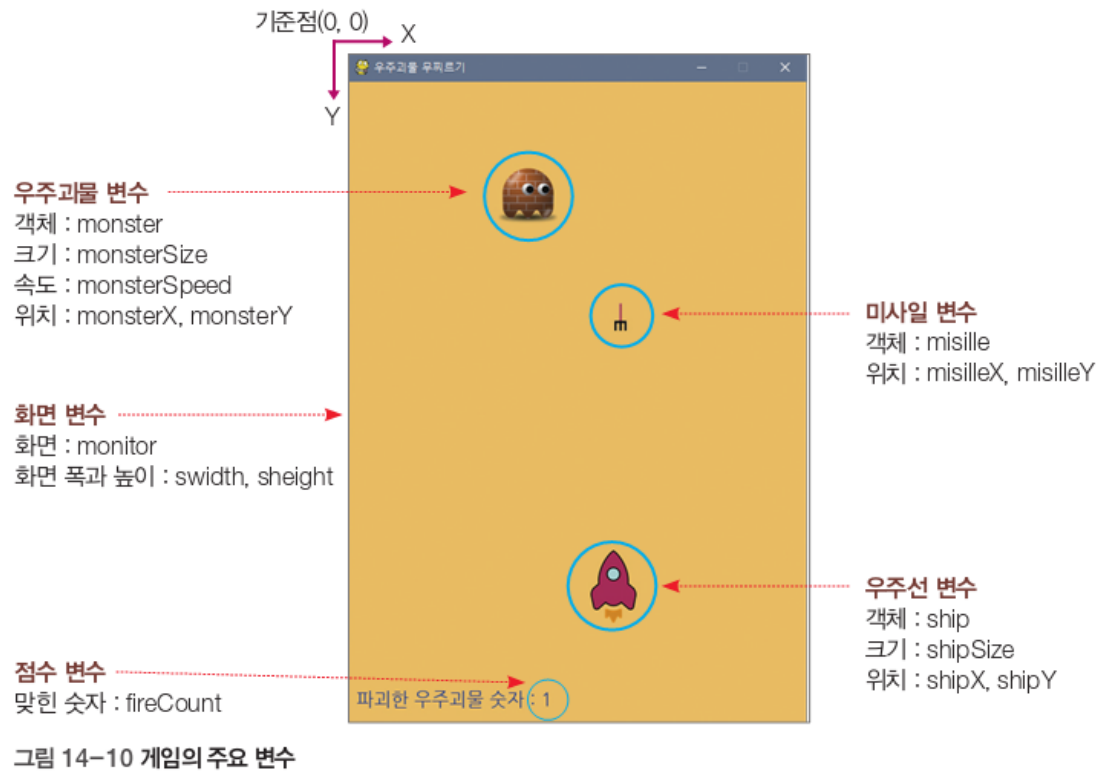
- 기능 4. 우주선에서 미사일 발사하기(code14-12.Py)



- 기능 4. 우주괴물 맞히고 점수 계산하기(code14-14.Py)



■ 주요 변수 소개



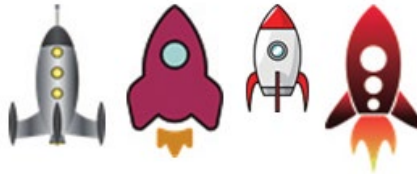


그림 14-11 우주선 이미지



그림 14-12 우주괴물 이미지



그림 14-13 미사일 이미지

■ 슈팅 게임 프로젝트의 완성

■ 기본 화면 구성하기

Code14-10.py

```
1  import pygame
2  import random
3  import sys
4
5
6  ## 함수 선언 부분 ##
7  # @기능 2-5 : 매개변수로 받은 객체를 화면에 그리는 함수를 선언한다.
8  # @기능 5-4 : 점수를 화면에 쓰는 함수를 선언한다.
9
10 def playGame() :
11     global monitor
12
13     r = random.randrange(0, 256)
14     g = random.randrange(0, 256)
15     b = random.randrange(0, 256)
16
17     # @기능 2-2 : 우주선의 초기 위치 키보드를 눌렀을 때 이동량을 저장할 변수를 선언한다.
18     # @기능 3-2 : 우주괴물을 무작위로 추출하고 크기와 위치를 설정한다.
19     # @기능 4-2 : 미사일 좌표를 초기화한다.
20     # @기능 5-1 : 맞힌 우주괴물 숫자를 저장할 변수를 선언한다.
21
22     # 무한 반복
23     while True :
```

```
24     (pygame.time.Clock()).tick(50)    # 게임 진행을 늦춘다(10~100 정도가 적당).
25     monitor.fill((r, g, b))          # 화면 배경을 칠한다.
26
27     # 키보드나 마우스 이벤트가 들어오는지 체크한다.
28     for e in pygame.event.get() :
29         if e.type in [pygame.QUIT] :
30             pygame.quit()
31             sys.exit()
32
33     # @기능 2-3 : 방향키에 따라 우주선이 움직이게 한다.
34     # @기능 4-3 : 스페이스바를 누르면 미사일을 발사한다.
35
36     # @기능 2-4 : 우주선이 화면 안에서만 움직이게 한다.
37     # @기능 3-3 : 우주괴물이 자동으로 나타나 왼쪽에서 오른쪽으로 움직인다.
38     # @기능 4-4 : 미사일을 화면에 표시한다.
39     # @기능 5-2 : 우주괴물이 미사일에 맞았는지 체크한다.
40     # @기능 5-3 : 점수를 화면에 쓰는 함수를 호출한다.
41
42     # 화면을 업데이트한다.
43     pygame.display.update()
44     print('~', end = '')
45
46
```

Section 02 슈팅 게임 프로젝트

```
47 ## 전역 변수 선언 부분 ##
48 r, g, b = [0] * 3                # 게임 배경색
49 swidth, sheight = 500, 700      # 화면 크기
50 monitor = None                  # 게임 화면
51
52 # @기능 3-1 : 무작위로 사용할 우주괴물 이미지를 10개 준비한다.
53
54
55 ## 메인 코드 부분 ##
56 pygame.init()
57 monitor = pygame.display.set_mode((swidth, sheight))
58 pygame.display.set_caption('우주괴물 무찌르기')
59
60 # @기능 2-1 : 우주선 이미지를 준비하고 크기를 구한다.
61 # @기능 4-1 : 미사일 이미지를 추가한다.
62
63 playGame()
```



- 우주선 이미지를 추가하고 방향키로 움직이기

Code14-11.py

```
import pygame
import random
import sys
```

```
## 함수 선언 부분 ##
```

```
# @기능 2-5 : 매개변수로 받은 객체를 화면에 그리는 함수를 선언한다.
```

```
def paintEntity(entity, x, y):
    monitor.blit(entity, (int(x), int(y)))
```

⑦

@기능 5-4 : 점수를 화면에 쓰는 함수를 선언한다.

def playGame() :

 global monitor, ship ❸

 r = random.randrange(0, 256)

 g = random.randrange(0, 256)

 b = random.randrange(0, 256)

@기능 2-2 : 우주선의 초기 위치 키보드를 눌렀을 때 이동량을 저장할 변수를 선언한다.

 shipX = swidth / 2 # 우주선 위치

 shipY = sheight * 0.8

 dx, dy = 0, 0 # 키보드를 누를 때 우주선의 이동량

❹

@기능 3-2 : 우주괴물을 무작위로 추출하고 크기와 위치를 설정한다.

@기능 4-2 : 미사일 좌표를 초기화한다.

@기능 5-1 : 맞춘 우주괴물 숫자를 저장할 변수를 선언한다.

무한 반복

while True :

 (pygame.time.Clock()).tick(50) # 게임 진행을 늦춘다(10~100 정도가 적당).

 monitor.fill((r, g, b)) # 화면 배경을 칠한다.

```
# 키보드나 마우스 이벤트가 들어오는지 체크한다.
```

```
for e in pygame.event.get() :
```

```
    if e.type in [pygame.QUIT] :
```

```
        pygame.quit()
```

```
        sys.exit()
```

```
# @기능 2-3 : 방향키에 따라 우주선이 움직이게 한다.
```

```
# 방향키를 누르면 우주선이 이동한다(누르고 있으면 계속 이동한다).
```

```
if e.type in [pygame.KEYDOWN] :
```

```
    if e.key == pygame.K_LEFT : dx = -5
```

```
    elif e.key == pygame.K_RIGHT : dx = +5
```

```
    elif e.key == pygame.K_UP : dy = -5
```

```
    elif e.key == pygame.K_DOWN : dy = +5
```

```
# @기능 4-3 : 스페이스바를 누르면 미사일을 발사한다.
```

```
# 방향키를 떼면 우주선이 멈춘다.
```

```
if e.type in [pygame.KEYUP] :
```

```
    if e.key == pygame.K_LEFT or e.key == pygame.K_RIGHT \
```

```
        or e.key == pygame.K_UP or e.key == pygame.K_DOWN : dx, dy = 0, 0
```

⑤

Section 02 슈팅 게임 프로젝트

```
# @기능 2-4 : 우주선이 화면 안에서만 움직이게 한다.
if (0 < shipX + dx and shipX + dx <= swidth - shipSize[0]) \
    and (sheight / 2 < shipY + dy and shipY + dy <= sheight - shipSize[1]) :
    shipX += dx
    shipY += dy
    paintEntity(ship, shipX, shipY) # 우주선을 화면에 표시한다.

# @기능 3-3 : 우주괴물이 자동으로 나타나 왼쪽에서 오른쪽으로 움직인다.
# @기능 4-4 : 미사일을 화면에 표시한다.
    # @기능 5-2 : 우주괴물이 미사일에 맞았는지 체크한다.
# @기능 5-3 : 점수를 화면에 쓰는 함수를 호출한다.

# 화면을 업데이트한다.
pygame.display.update()
print('~', end = ' ')

## 전역 변수 선언 부분 ##
r, g, b = [0] * 3 # 게임 배경색
swidth, sheight = 500, 700 # 화면 크기
monitor = None # 게임 화면
ship, shipSize = None, 0 # 우주선의 객체와 크기 변수 ❶
```

Section 02 슈팅 게임 프로젝트

@기능 3-1 : 무작위로 사용할 우주괴물 이미지를 10개 준비한다.

메인 코드 부분

```
pygame.init()
monitor = pygame.display.set_mode((swidth, sheight))
pygame.display.set_caption('우주괴물 무찌르기')
```

@기능 2-1 : 우주선 이미지를 준비하고 크기를 구한다.

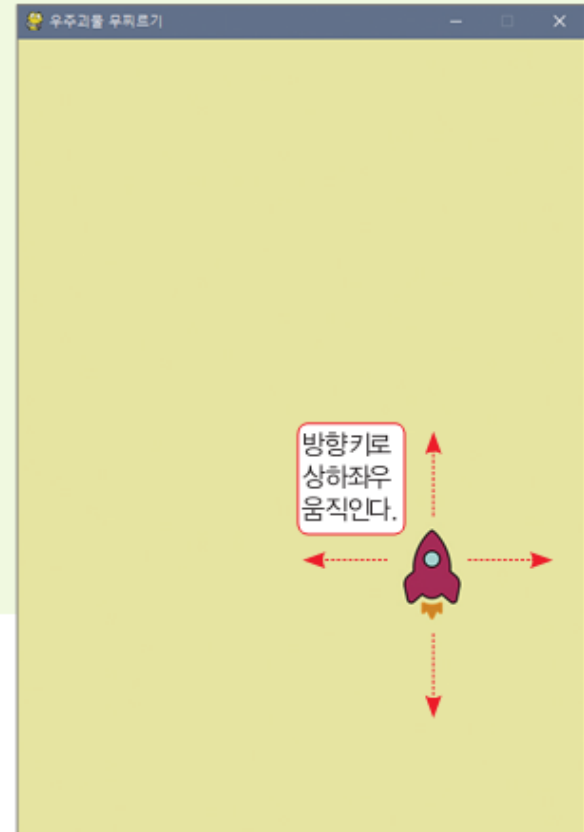
```
ship = pygame.image.load('ship02.png')
```

```
shipSize = ship.get_rect().size
```

②

@기능 4-1 : 미사일 이미지를 추가한다.

```
playGame()
```



- 우주괴물이 나타나면 자동으로 움직이기

Code14-12.py

```
import pygame
import random
import sys

## 함수 선언 부분 ##
# @기능 2-5 : 매개변수로 받은 객체를 화면에 그리는 함수를 선언한다.
def paintEntity(entity, x, y) :
    monitor.blit(entity, (int(x), int(y)))

# @기능 5-4 : 점수를 화면에 쓰는 함수를 선언한다.

def playGame() :
    global monitor, ship, monster ❷

    r = random.randrange(0, 256)
    g = random.randrange(0, 256)
    b = random.randrange(0, 256)
```

Section 02 슈팅 게임 프로젝트

@기능 2-2 : 우주선의 초기 위치 키보드를 눌렀을 때 이동량을 저장할 변수를 선언한다.

shipX = swidth / 2 # 우주선 위치

shipY = sheight * 0.8

dx, dy = 0, 0 # 키보드를 누를 때 우주선의 이동량

@기능 3-2 : 우주괴물을 무작위로 추출하고 크기와 위치를 설정한다.

monster = pygame.image.load(random.choice(monsterImage))

monsterSize = monster.get_rect().size # 우주괴물 크기

monsterX = 0

monsterY = random.randrange(0, int(swidth * 0.3)) # 상위 30% 위치까지만

monsterSpeed = random.randrange(1, 5)

3

@기능 4-2 : 미사일 좌표를 초기화한다.

@기능 5-1 : 맞춘 우주괴물 숫자를 저장할 변수를 선언한다.

무한 반복

while True :

 (pygame.time.Clock()).tick(50) # 게임 진행을 늦춘다(10~100 정도가 적당).

 monitor.fill((r, g, b)) # 화면 배경을 칠한다.

 # 키보드나 마우스 이벤트가 들어오는지 체크한다.

 for e in pygame.event.get() :

 if e.type in [pygame.QUIT] :

 pygame.quit()

 sys.exit()

```
# @기능 2-3 : 방향키에 따라 우주선이 움직이게 한다.
# 방향키를 누르면 우주선이 이동한다(누르고 있으면 계속 이동한다).
if e.type in [pygame.KEYDOWN]:
    if e.key == pygame.K_LEFT : dx = -5
    elif e.key == pygame.K_RIGHT : dx = +5
    elif e.key == pygame.K_UP : dy = -5
    elif e.key == pygame.K_DOWN : dy = +5
    # @기능 4-3 : 스페이스바를 누르면 미사일을 발사한다.

# 방향키를 떼면 우주선이 멈춘다.
if e.type in [pygame.KEYUP]:
    if e.key == pygame.K_LEFT or e.key == pygame.K_RIGHT \
        or e.key == pygame.K_UP or e.key == pygame.K_DOWN : dx, dy = 0, 0

# @기능 2-4 : 우주선이 화면 안에서만 움직이게 한다.
if (0 < shipX + dx and shipX + dx <= swidth - shipSize[0]) \
    and (sheight / 2 < shipY + dy and shipY + dy <= sheight - shipSize[1]):
    # 화면의 중앙까지만

    shipX += dx
    shipY += dy
```

```
paintEntity(ship, shipX, shipY)    # 우주선을 화면에 표시한다.
```

```
# @기능 3-3 : 우주괴물이 자동으로 나타나 왼쪽에서 오른쪽으로 움직인다.
```

```
monsterX += monsterSpeed
```

```
if monsterX > swidth :
```

```
    monsterX = 0
```

```
    monsterY = random.randrange(0, int(swidth * 0.3))
```

```
    # 우주괴물 이미지를 무작위로 선택한다.
```

```
    monster = pygame.image.load(random.choice(monsterImage))
```

```
    monsterSize = monster.get_rect().size
```

```
    monsterSpeed = random.randrange(1, 5)
```

```
paintEntity(monster, monsterX, monsterY)
```

```
# @기능 4-4 : 미사일을 화면에 표시한다.
```

```
    # @기능 5-2 : 우주괴물이 미사일에 맞았는지 체크한다.
```

```
# @기능 5-3 : 점수를 화면에 쓰는 함수를 호출한다.
```

```
# 화면을 업데이트한다.
```

```
pygame.display.update()
```

4

Section 02 슈팅 게임 프로젝트

```
## 전역 변수 선언 부분 ##
r, g, b = [0] * 3          # 게임 배경색
swidth, sheight = 500, 700 # 화면 크기
monitor = None             # 게임 화면
ship, shipSize = None, 0   # 우주선의 객체와 크기 변수
```

@기능 3-1 : 무작위로 사용할 우주괴물 이미지를 10개 준비한다.

```
monsterImage = ['monster01.png', 'monster02.png', 'monster03.png', 'monster04.png', \
                'monster05.png', 'monster06.png', 'monster07.png', 'monster08.png', \
                'monster09.png', 'monster10.png']
```

monster = None # 우주괴물

```
## 메인 코드 부분 ##
pygame.init()
monitor = pygame.display.set_mode((swidth, sheight))
pygame.display.set_caption('우주괴물 무찌르기')
```

@기능 2-1 : 우주선 이미지를 준비하고 크기를 구한다.

```
ship = pygame.image.load('ship02.png')
shipSize = ship.get_rect().size
```

@기능 4-1 : 미사일 이미지를 추가한다.

```
playGame()
```



- 우주선에서 미사일 발사하기

Code14-13.py

```
import pygame
import random
import sys

## 함수 선언 부분 ##
# @기능 2-5 : 매개변수로 받은 객체를 화면에 그리는 함수를 선언한다.
def paintEntity(entity, x, y):
```



```
monitor.blit(entity, (int(x), int(y)))
```

@기능 5-4 : 점수를 화면에 쓰는 함수를 선언한다.

```
def playGame() :
```

```
    global monitor, ship, monster, missile ❷
```

```
    r = random.randrange(0, 256)
```

```
    g = random.randrange(0, 256)
```

```
    b = random.randrange(0, 256)
```

@기능 2-2 : 우주선의 초기 위치 키보드를 눌렀을 때 이동량을 저장할 변수를 선언한다.

```
    shipX = swidth / 2      # 우주선 위치
```

```
    shipY = sheight * 0.8
```

```
    dx, dy = 0, 0          # 키보드를 누를 때 우주선의 이동량
```

@기능 3-2 : 우주괴물을 무작위로 추출하고 크기와 위치를 설정한다.

```
    monster = pygame.image.load(random.choice(monsterImage))
```

```
    monsterSize = monster.get_rect().size          # 우주괴물 크기
```

```
    monsterX = 0
```

```
    monsterY = random.randrange(0, int(swidth * 0.3))  # 상위 30% 위치까지만
```

```
    monsterSpeed = random.randrange(1, 5)
```

```
# @기능 4-2 : 미사일 좌표를 초기화한다.
missileX, missileY = None, None      # None은 미사일을 쏘지 않았다는 의미이다.
# @기능 5-1 : 맞춘 우주괴물 숫자를 저장할 변수를 선언한다.

# 무한 반복
while True :
    (pygame.time.Clock()).tick(50)    # 게임 진행을 늦춘다(10~100 정도가 적당).
    monitor.fill((r, g, b))           # 화면 배경을 칠한다.

    # 키보드나 마우스 이벤트가 들어오는지 체크한다.
    for e in pygame.event.get() :
        if e.type in [pygame.QUIT] :
            pygame.quit()
            sys.exit()

    # @기능 2-3 : 방향키에 따라 우주선이 움직이게 한다.
    # 방향키를 누르면 우주선이 이동한다(누르고 있으면 계속 이동한다).
    if e.type in [pygame.KEYDOWN] :
        if e.key == pygame.K_LEFT : dx = -5
        elif e.key == pygame.K_RIGHT : dx = +5
        elif e.key == pygame.K_UP : dy = -5
        elif e.key == pygame.K_DOWN : dy = +5
```

```
# @기능 4-3 : 스페이스바를 누르면 미사일을 발사한다.
elif e.key == pygame.K_SPACE :
    if missileX == None :      # 미사일을 쏜 적이 없다면
        missileX = shipX + shipSize[0] / 2
        # 우주선 위치에서 미사일을 발사한다.
        missileY = shipY

# 방향키를 떼면 우주선이 멈춘다.
if e.type in [pygame.KEYUP] :
    if e.key == pygame.K_LEFT or e.key == pygame.K_RIGHT \
        or e.key == pygame.K_UP or e.key == pygame.K_DOWN : dx, dy = 0, 0

# @기능 2-4 : 우주선이 화면 안에서만 움직이게 한다.
if (0 < shipX + dx and shipX + dx <= swidth - shipSize[0]) \
    and (sheight / 2 < shipY + dy and shipY + dy <= sheight - shipSize[1]) :
    # 화면의 중앙까지만

    shipX += dx
    shipY += dy
paintEntity(ship, shipX, shipY)      # 우주선을 화면에 표시한다.
```

Section 02 슈팅 게임 프로젝트

@기능 3-3 : 우주괴물이 자동으로 나타나 왼쪽에서 오른쪽으로 움직인다.

monsterX += monsterSpeed

if monsterX > swidth :

 monsterX = 0

 monsterY = random.randrange(0, int(swidth * 0.3))

 # 우주괴물 이미지를 무작위로 선택한다.

 monster = pygame.image.load(random.choice(monsterImage))

 monsterSize = monster.get_rect().size

 monsterSpeed = random.randrange(1, 5)

paintEntity(monster, monsterX, monsterY)

@기능 4-4 : 미사일을 화면에 표시한다.

if missileX != None : # 총알을 쏘면 좌표를 위로 변경한다.

 missileY -= 10

 if missileY < 0 :

 missileX, missileY = None, None # 총알이 사라진다.

if missileX != None : # 미사일을 쏜 적이 있으면 미사일을 그려 준다.

 paintEntity(missile, missileX, missileY)

 # @기능 5-2 : 우주괴물이 미사일에 맞았는지 체크한다.

@기능 5-3 : 점수를 화면에 쓰는 함수를 호출한다.

화면을 업데이트한다.

pygame.display.update()

6

```
## 전역 변수 선언 부분 ##
r, g, b = [0] * 3                # 게임 배경색
swidth, sheight = 500, 700       # 화면 크기
monitor = None                   # 게임 화면
ship, shipSize = None, 0         # 우주선의 객체와 크기 변수

# @기능 3-1 : 무작위로 사용할 우주괴물 이미지를 10개 준비한다.
monsterImage = ['monster01.png', 'monster02.png', 'monster03.png', 'monster04.png', \
                'monster05.png', 'monster06.png', 'monster07.png', 'monster08.png', \
                'monster09.png', 'monster10.png']
monster = None                   # 우주괴물

missile = None                   # 미사일 ❶

## 메인 코드 부분 ##
pygame.init()
monitor = pygame.display.set_mode((swidth, sheight))
pygame.display.set_caption('우주괴물 무찌르기')
```

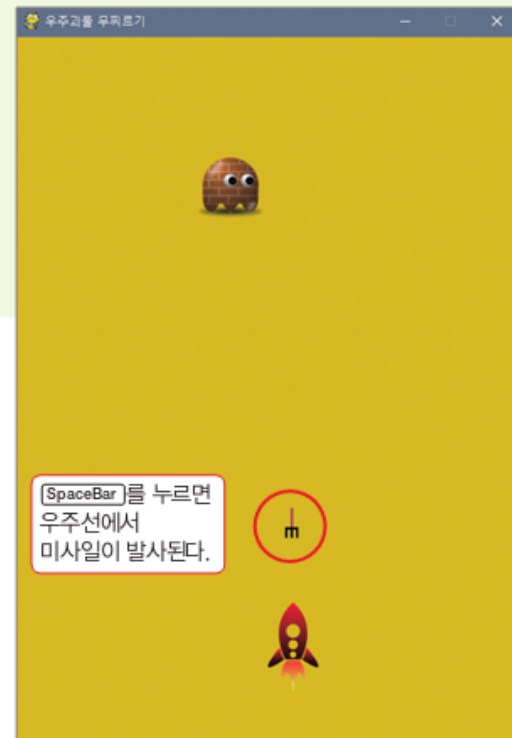
Section 02 슈팅 게임 프로젝트

```
## 메인 코드 부분 ##
pygame.init()
monitor = pygame.display.set_mode((swidth, sheight))
pygame.display.set_caption('우주괴물 무찌르기')

# @기능 2-1 : 우주선 이미지를 준비하고 크기를 구한다.
ship = pygame.image.load('ship02.png')
shipSize = ship.get_rect().size

# @기능 4-1 : 미사일 이미지를 추가한다.
missile = pygame.image.load('missile.png')

playGame()
```



- 우주괴물 맞히고 점수 계산하기([프로젝트 2]의 완성)

Code14-14.py

```
import pygame
import random
import sys

## 함수 선언 부분 ##
# @기능 2-5 : 매개변수로 받은 객체를 화면에 그리는 함수를 선언한다.
def paintEntity(entity, x, y) :
    monitor.blit(entity, (int(x), int(y)))

# @기능 5-4 : 점수를 화면에 쓰는 함수를 선언한다.
def writeScore(score) :
    myfont = pygame.font.Font('NanumGothic.ttf', 20) # 한글 폰트
    txt = myfont.render(u'파괴한 우주괴물 수 : ' + str(score), True, (255-r, 255-g, 255-b))
    monitor.blit(txt, (10, sheight - 40))

def playGame() :
    global monitor, ship, monster, missile

    r = random.randrange(0, 256)
    g = random.randrange(0, 256)
    b = random.randrange(0, 256)
```

4

Section 02 슈팅 게임 프로젝트

```
# @기능 2-2 : 우주선의 초기 위치 키보드를 눌렀을 때 이동량을 저장할 변수를 선언한다.
shipX = swidth / 2      # 우주선 위치
shipY = sheight * 0.8
dx, dy = 0, 0          # 키보드를 누를 때 우주선의 이동량

# @기능 3-2 : 우주괴물을 무작위로 추출하고 크기와 위치를 설정한다.
monster = pygame.image.load(random.choice(monsterImage))
monsterSize = monster.get_rect().size          # 우주괴물 크기
monsterX = 0
monsterY = random.randrange(0, int(swidth * 0.3)) # 상위 30% 위치까지만
monsterSpeed = random.randrange(1, 5)

# @기능 4-2 : 미사일 좌표를 초기화한다.
missileX, missileY = None, None                # None은 미사일을 쏘지 않았다는 의미이다.

# @기능 5-1 : 맞춘 우주괴물 숫자를 저장할 변수를 선언한다.
fireCount = 0 ] ❶

# 무한 반복
while True :
    (pygame.time.Clock()).tick(50)      # 게임 진행을 늦춘다(10~100 정도가 적당).
    monitor.fill((r, g, b))            # 화면 배경을 칠한다.
```



```
# 키보드나 마우스 이벤트가 들어오는지 체크한다.
for e in pygame.event.get() :
    if e.type in [pygame.QUIT] :
        pygame.quit()
        sys.exit()

# @기능 2-3 : 방향키에 따라 우주선이 움직이게 한다.
# 방향키를 누르면 우주선이 이동한다(누르고 있으면 계속 이동한다).
if e.type in [pygame.KEYDOWN] :
    if e.key == pygame.K_LEFT : dx = -5
    elif e.key == pygame.K_RIGHT : dx = +5
    elif e.key == pygame.K_UP : dy = -5
    elif e.key == pygame.K_DOWN : dy = +5
    # @기능 4-3 : 스페이스바를 누르면 미사일을 발사한다.
    elif e.key == pygame.K_SPACE :
        if missileX == None :    # 미사일을 쏜 적이 없다면
            missileX = shipX + shipSize[0] / 2
            # 우주선 위치에서 미사일을 발사한다.
            missileY = shipY
```

```
# 방향키를 떼면 우주선이 멈춘다.
if e.type in [pygame.KEYUP]:
    if e.key == pygame.K_LEFT or e.key == pygame.K_RIGHT \
        or e.key == pygame.K_UP or e.key == pygame.K_DOWN : dx, dy = 0, 0

# @기능 2-4 : 우주선이 화면 안에서만 움직이게 한다.
if (0 < shipX + dx and shipX + dx <= swidth - shipSize[0]) \
    and (sheight / 2 < shipY + dy and shipY + dy <= sheight - shipSize[1]) :
    # 화면의 중앙까지만

    shipX += dx
    shipY += dy
paintEntity(ship, shipX, shipY)    # 우주선을 화면에 표시한다.

# @기능 3-3 : 우주괴물이 자동으로 나타나 왼쪽에서 오른쪽으로 움직인다.
monsterX += monsterSpeed
if monsterX > swidth :
    monsterX = 0
    monsterY = random.randrange(0, int(swidth * 0.3))
    # 우주괴물 이미지를 무작위로 선택한다.
    monster = pygame.image.load(random.choice(monsterImage))
    monsterSize = monster.get_rect().size
    monsterSpeed = random.randrange(1, 5)

paintEntity(monster, monsterX, monsterY)
```

```
# @기능 4-4 : 미사일을 화면에 표시한다.
if missileX != None :      # 총알을 쏘면 좌표를 위로 변경한다.
    missileY -= 10
    if missileY < 0 :
        missileX, missileY = None, None    # 총알이 사라진다.
if missileX != None :      # 미사일을 쏜 적이 있으면 미사일을 그려 준다.
    paintEntity(missile, missileX, missileY)
# @기능 5-2 : 우주괴물이 미사일에 맞았는지 체크한다.
if (monsterX < missileX and missileX < monsterX + monsterSize[0]) and \
    (monsterY < missileY and missileY < monsterY + monsterSize[1]) :
    fireCount += 1

    # 우주괴물을 초기화한다(무작위 이미지로 다시 준비한다).
    monster = pygame.image.load(random.choice(monsterImage))
    monsterSize = monster.get_rect().size
    monsterX = 0
    monsterY = random.randrange(0, int(swidth * 0.3))
    monsterSpeed = random.randrange(1, 5)

    # 미사일을 초기화한다.
    missileX, missileY = None, None    # 총알이 사라진다.
```

2

```
# @기능 5-3 : 점수를 화면에 쓰는 함수를 호출한다.
```

3

```
writeScore(fireCount)
```

```
# 화면을 업데이트한다.
```

```
pygame.display.update()
```

```
## 전역 변수 선언 부분 ##
```

```
r, g, b = [0] * 3          # 게임 배경색
```

```
swidth, sheight = 500, 700 # 화면 크기
```

```
monitor = None             # 게임 화면
```

```
ship, shipSize = None, 0   # 우주선의 객체와 크기 변수
```

```
# @기능 3-1 : 무작위로 사용할 우주괴물 이미지를 10개 준비한다.
```

```
monsterImage = ['monster01.png', 'monster02.png', 'monster03.png', 'monster04.png', \
                'monster05.png', 'monster06.png', 'monster07.png', 'monster08.png', \
                'monster09.png', 'monster10.png']
```

```
monster = None             # 우주괴물
```

```
missile = None            # 미사일
```

Section 02 슈팅 게임 프로젝트

```
## 메인 코드 부분 ##  
pygame.init()  
monitor = pygame.display.set_mode((swidth, sheight))  
pygame.display.set_caption('우주괴물 무찌르기')  
  
# @기능 2-1 : 우주선 이미지를 준비하고 크기를 구한다.  
ship = pygame.image.load('ship02.png')  
shipSize = ship.get_rect().size  
  
# @기능 4-1 : 미사일 이미지를 추가한다.  
missile = pygame.image.load('missile.png')  
  
playGame()
```





Thank You

