



קריית החינוך
פארק המדע
בית לערכים
למציאות ולחדשנות

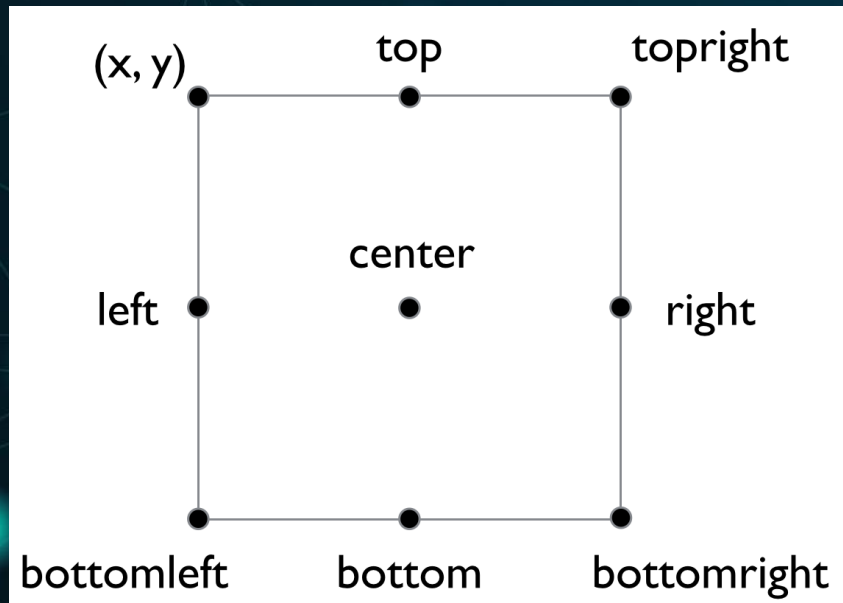
PyGame Rect

גלעד מרקמן



PyGame.rect

- Rect – הוא אובייקט שנועד לשמור קואורדינטות ריבועיות.
- האובייקט מייצג מסגרת של משטח (surface) מסויים.
- אובייקט ה rect עוזר לנו למקם משטחים (כולל תמונות) במסך.
- אובייקט ה rect אינו ניתן להצגה על המסך, אלא זה אובייקט עזר להציג משטחים אחרים.



יצירת אובייקט Rect

- הדרך הנוחה ביותר ליצור אובייקט rect היא באמצעות המשטח (התמונה) שאותה אנחנו רוצים למקם במסך.

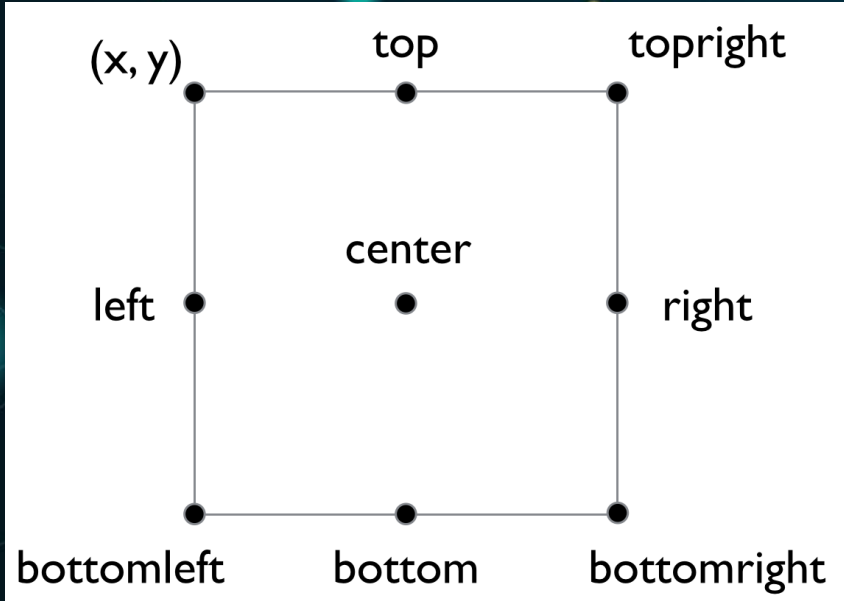
```
space_ship = pygame.image.load("img/spacecraft.png")  
space_ship = pygame.transform.scale(space_ship, (40, 50))  
space_ship_rect = space_ship.get_rect()
```

• לדוגמה:

- בדרך זו יצרנו אובייקט rect המכונה `space_ship_rect`. באמצעותו נוכל למקם בקלות את התמונה של ספינת החלל.

המאפיינים של אובייקט Rect

• אובייקט rect כולל את המאפיינים הבאים:



• x, y

• top, left, bottom, right

• topleft, bottomleft, topright, bottomright

• midtop, midleft, midbottom, midright

• center, centerx, centery

• size, width, height

• w, h

• ניתן לשנות את כל אחד מהמאפיינים באמצעות הצבת ערך:

```
space_ship_rect.center = x, y
```

פעולות של אובייקט Rect

- אובייקט rect כולל גם מספר פעולות, כגון:
- move - הזזה של כל הקורדינטות מספר קבוע של פיקסלים. הפקודה תשנה את כל המאפיינים בהתאם (...center, top, right)

```
space_ship_rect = space_ship_rect.move(x, y)
```

- Move_ip – כמו move אבל פועלת על האובייקט ולא מחזירה אובייקט חדש.

```
space_ship_rect.move_ip(x,y)
```

- Colliserect – טיפל בהתנגדויות בין משטחים. ראה בהמשך.

• רשימת פעולות של אובייקט Rect

שימוש באובייקט Rect

- השימוש באובייקט יכול להיעשות, למשל, באמצעות פקודת `.blit`. האובייקט משמש אותנו במקום הקואורדינטות.

```
space_ship_rect = space_ship.get_rect()  
space_ship_rect.center = x, y  
screen.blit(space_ship_rotated, space_ship_rect)
```

- בדוגמה זו הדפסנו על המסך את החללית במיקום התואם ל `.rect` לפני כן, עדכנו את ה `.rect`.

- ניתן להשתמש ב `rect` לבירור מיקום האובייקט שלנו, לדוגמה:

```
if rect.bottom >= HEIGHT:  
    speed_y = -2
```

הדגמה – rect1

פרויקט ב GitHub

```
import pygame
from Graphics import *
pygame.init()

screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption('Reversi')
clock = pygame.time.Clock()
screen.fill(LIGHTGRAY)

x, y = 50, 50
dx, dy = 1, 1

space_ship = pygame.image.load("img/spacecraft.png")
space_ship = pygame.transform.scale(space_ship, (40, 50))
space_ship_rect = space_ship.get_rect()
```

```
run = True
while (run):
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

    screen.fill(LIGHTGRAY)

    x = (x + dx) % WIDTH
    y = (y + dy) % HEIGHT
    space_ship_rect.center = x, y
    screen.blit(space_ship, space_ship_rect)
    pygame.display.flip()
    clock.tick(FPS)

pygame.quit()
```

הדגמה – rect2

פרוייקט ב GitHub

```
import pygame
from Graphics import *
pygame.init()

screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption('Reversi')
clock = pygame.time.Clock()
screen.fill(LIGHTGRAY)

x, y = 250, 350
speed_x, speed_y = 1, 1

rectangle = pygame.Surface((100,80))
rectangle.fill('red')
rect = rectangle.get_rect(center = (x,y))
```

```
run = True
while (run):
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

    screen.fill(LIGHTGRAY)

    if rect.bottom >= HEIGHT:
        speed_y = -2
    elif rect.top <= 0:
        speed_y = 2
    if rect.right >= WIDTH:
        speed_x = -2
    elif rect.left <= 0:
        speed_x = 2

    # rect.move_ip(speed_x, speed_y)
    rect = rect.move(speed_x, speed_y)
    screen.blit(rectangle, rect)
    pygame.display.flip()
    clock.tick(FPS)

pygame.quit()
```