

```

1 import math
2 import random
3
4
5 def app_started(app):
6     app.timer_delay = 30
7     app.pressed_keys = set()
8     init(app)
9
10
11 def init(app):
12     '''
13     Initialize game state. Called when game is started or restarted. Will
14     create all necessary variables and set the game state to 'running'.
15     '''
16     app.game_state = 'running'
17
18     app.spaceship_x = app.width/2
19     app.spaceship_y = app.height/2
20     app.spaceship_radius = 10
21     app.angle = 0
22     app.speed_x = 0.3
23     app.speed_y = 0.2
24
25     app.bullets = [
26         # bullets have the form (x, y, angle), list is initially empty
27     ]
28
29     app.stones = [
30         # (x, y, radius, speed_x, speed_y)
31         (100, 100, 20, 1, 1),
32         (200, 300, 30, -1, 1),
33         (300, 300, 40, -1, -1),
34         (600, 400, 50, 1, -1)
35     ]
36
37
38 def key_pressed(app, event):
39     app.pressed_keys.add(event.key)
40     if app.game_state == 'game_over' and event.key in ['Return', 'Enter']:
41         init(app)
42     if event.key == 'Space':
43         app.bullets.append((app.spaceship_x, app.spaceship_y, app.angle))
44
45
46 def key_released(app, event):
47     app.pressed_keys.remove(event.key)
48
49
50 def timer_fired(app):
51     move_spaceship(app)
52     do_spaceship_steering(app)
53     do_move_bullets(app)
54     do_move_stones(app)
55     check_bullet_hits(app)
56     check_stone_collision(app)
57     randomly_add_stones(app)
58
59
60 def move_spaceship(app):
61     app.spaceship_x += app.speed_x
62     app.spaceship_y += app.speed_y
63
64     # Wrap around
65     if app.spaceship_x < -app.spaceship_radius:
66         app.spaceship_x = app.width + app.spaceship_radius
67     elif app.spaceship_x > app.width + app.spaceship_radius:
68         app.spaceship_x = -app.spaceship_radius
69     if app.spaceship_y < -app.spaceship_radius:
70         app.spaceship_y = app.height + app.spaceship_radius
71     elif app.spaceship_y > app.height + app.spaceship_radius:
72         app.spaceship_y = -app.spaceship_radius
73
74
75 def do_spaceship_steering(app):
76     # Rotate spaceship
77     if 'Left' in app.pressed_keys:
78         app.angle += 0.1
79     elif 'Right' in app.pressed_keys:
80         app.angle -= 0.1
81
82     # Accelerate spaceship
83     elif 'Up' in app.pressed_keys:
84         app.speed_x += 0.1 * math.cos(app.angle)
85         app.speed_y += 0.1 * math.sin(app.angle)
86
87
88 def do_move_stones(app):
89     for i in range(len(app.stones)):
90         x, y, radius, speed_x, speed_y = app.stones[i]
91         x += speed_x
92         y += speed_y
93
94         # Wrap around
95         if x < -radius:
96             x = app.width + radius
97         elif x > app.width + radius:
98             x = -radius
99
100         if y < -radius:
101             y = app.height + radius
102         elif y > app.height + radius:
103             y = -radius
104
105         app.stones[i] = (x, y, radius, speed_x, speed_y)
106
107
108 def do_move_bullets(app):
109     # Move bullets
110     for i in range(len(app.bullets)):
111         x, y, angle = app.bullets[i]
112         x += 5 * math.cos(angle)
113         y -= 5 * math.sin(angle)
114         app.bullets[i] = (x, y, angle)
115
116     # Keep only bullets that are within the screen
117     new_bullets = []
118     for x, y, angle in app.bullets:
119         if 0 <= x <= app.width and 0 <= y <= app.height:
120             new_bullets.append((x, y, angle))
121     app.bullets = new_bullets
122
123
124 def check_bullet_hits(app):
125     '''
126     Check if any of the bullets have hit a stone. If so, remove the
127     bullet and the stone.
128     '''
129     for i in range(len(app.bullets)):
130         bullet_x, bullet_y, _ = app.bullets[i]
131         for j in range(len(app.stones)):
132             x, y, radius, _, _ = app.stones[j]
133             distance = math.sqrt((x - bullet_x)**2 + (y - bullet_y)**2)
134             if distance < radius:
135                 app.bullets.pop(i)
136                 app.stones.pop(j)
137
138
139 def check_stone_collision(app):
140     for stone in app.stones:
141         x, y, radius, _, _ = stone
142         dist = math.sqrt((x - app.spaceship_x)**2 + (y - app.spaceship_y)**2)
143         if dist < radius + app.spaceship_radius:
144             app.game_state = 'game_over'
145
146
147 def randomly_add_stones(app):
148     if random.random() < 0.01:
149         x = random.randrange(0, app.width)
150         y = random.randrange(0, app.height)
151         radius = random.randrange(10, 50)
152         speed_x = random.uniform(-1, 1)
153         speed_y = random.uniform(-1, 1)
154         # Move to the edge of the screen according to movement direction
155         if random.choice([True, False]):
156             x = -radius if speed_x > 0 else app.width + radius
157         else:
158             y = -radius if speed_y > 0 else app.height + radius
159
160         app.stones.append((x, y, radius, speed_x, speed_y))
161
162
163 def redraw_all(app, canvas):
164     canvas.create_rectangle(0, 0, app.width, app.height, fill='black')
165
166     if app.game_state == 'game_over':
167         canvas.create_text(
168             app.width/2, app.height/2,
169             text='Game Over', fill='white', font='Arial 50'
170         )
171     else:
172         draw_spaceship(app, canvas)
173         draw_bullets(app, canvas)
174         draw_stones(app, canvas)
175
176
177 def draw_spaceship(app, canvas):
178     x = app.spaceship_x
179     y = app.spaceship_y
180     r = app.spaceship_radius
181     angle = app.angle
182
183     # Find top in triangle based on angle
184     x_tip = x + r * 1.5 * math.cos(angle)
185     y_tip = y - r * 1.5 * math.sin(angle)
186     x_left = x + r * math.cos(angle + 2/3 * math.pi)
187     y_left = y - r * math.sin(angle + 2/3 * math.pi)
188     x_right = x + r * math.cos(angle - 2/3 * math.pi)
189     y_right = y - r * math.sin(angle - 2/3 * math.pi)
190
191     canvas.create_polygon(
192         x_tip, y_tip,
193         x_left, y_left,
194         x_right, y_right,
195         outline='white'
196     )
197
198     if 'Up' in app.pressed_keys:
199         draw_spaceship_flare(app, canvas, x, y, r, angle)
200
201
202 def draw_spaceship_flare(app, canvas, x, y, r, angle):
203     flare_l1_x = x + r * math.cos(angle + 0.9 * math.pi)
204     flare_l1_y = y - r * math.sin(angle + 0.9 * math.pi)
205     flare_l2_x = x + r * 1.5 * math.cos(angle + 0.9 * math.pi)
206     flare_l2_y = y - r * 1.5 * math.sin(angle + 0.9 * math.pi)
207     canvas.create_line(
208         flare_l1_x, flare_l1_y,
209         flare_l2_x, flare_l2_y,
210         fill='white'
211     )
212
213     flare_r1_x = x + r * math.cos(angle - 0.9 * math.pi)
214     flare_r1_y = y - r * math.sin(angle - 0.9 * math.pi)
215     flare_r2_x = x + r * 1.5 * math.cos(angle - 0.9 * math.pi)
216     flare_r2_y = y - r * 1.5 * math.sin(angle - 0.9 * math.pi)
217     canvas.create_line(
218         flare_r1_x, flare_r1_y,
219         flare_r2_x, flare_r2_y,
220         fill='white'
221     )
222
223
224 def draw_stones(app, canvas):
225     for x, y, r, speed_x, speed_y in app.stones:
226         canvas.create_oval(x-r, y-r, x+r, y+r, outline='white')
227
228
229 def draw_bullets(app, canvas):
230     for x, y, angle in app.bullets:
231         canvas.create_oval(x-2, y-2, x+2, y+2, fill='white', outline='')
232
233
234 if __name__ == '__main__':
235     from uib_inf100_graphics.event_app import run_app
236     run_app(width=800, height=600, title='INF100 Asteroids')
237

```