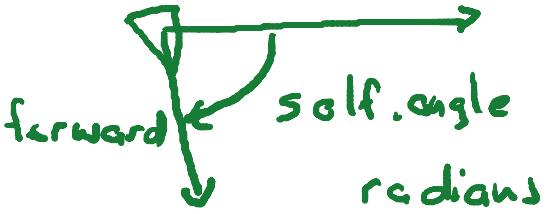
Controlling the spaceship's orientation

Spaceship class - two fields
- self.angle - ship orientation (scalar/float)
- self.angle_vel - ship's angular velocity (scalar/float)



Update method
- self.angle += self.angle_vel

Key handler controls self.angle_vel

Draw method
- canvas.draw_image(self.image, …, …, …, …, self.angle)

Relating position, velocity, and acceleration

Basic physics

- position - point
- velocity - vector
- acceleration - vector

Position update

position += velocity

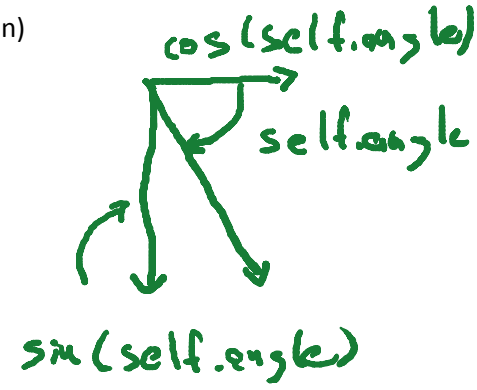
Velocity update

velocity += acceleration

Adding acceleration to the spaceship

Ship class - four fields

- self.pos - ship's position (vector/pair of floats)
- self.vel - ship's velocity (vector/pair of floats)
- self.angle - ship's orientation (scalar/float)
- self.thrust - whether ship is accelerating in forward direction (Boolean)




```
# Position update
self.pos[0] += self.vel[0]
self.pos[1] += self.vel[1]

# Velocity update - acceleration in direction of forward vector
forward = [math.cos(self.angle), math.sin(self.angle)]

if self.thrust:
    self.vel[0] += forward[0]
    self.vel[1] += forward[1]
```

← multiply by constant

Adding friction to the spaceship

Friction - let c be a small constant
friction = - c * velocity

acceleration = thrust + friction

velocity = velocity + acceleration
velocity = velocity + thrust + friction
velocity = velocity + thrust - c * velocity
velocity = (1 - c) * velocity + thrust

```
#Position update
self.pos[0] += self.vel[0]
self.pos[1] += self.vel[1]

#Friction udpate
self.vel[0] *= (1 - c)
self.vel[1] *= (1 - c)

#Thrust update - acceleration in direction of forward vector
forward = [math.cos(self.angle), math.sin(self.angle)]
if self.thrust:
    self.vel[0] += forward[0]
    self.vel[1] += forward[1]
```