Sandbox | CMU CS Academy

```
# HOW TO USE:
"""Try pressing the enter key breifly
Hold the enter key until the tachometer stops increasing
Try pressing the spacebar/ holding the spacebar"""
# Some setup
app.stepsPerSecond = 60
app.background='grey'
# Beginning of drawing Tachometer components
# Covers and dots
Circle(200,200,200,fill=rgb(25, 25, 25))
Circle(200,55,16,fill=rgb(45, 45, 45))
Circle(138,74,4,fill=rgb(45, 45, 45))
# Draw MPG Limit
Arc(200,55,154,154,217,26,fill=None,border=rgb(225, 225, 0),borderWidth=5)
Arc(200,55,134,134,217,26,fill=None,border=rgb(225, 225, 0),borderWidth=5)
Arc(200,55,124,124,215,29,fill=rgb(25, 25, 25))
# Draw smaller tick marks with getPointInDir and Line
for i in range(32):
    if (i % 5 != 0):
        outerX, outerY = getPointInDir(200, 200, 44+i*7.2, 192)
        innerX, innnerY = getPointInDir(200, 200, 44+i*7.2, 170)
        Line(outerX, outerY,innerX, innnerY,fill=rgb(225, 225, 0),lineWidth=5)
# Draw larger tick marks with getPointInDir and Line
for i in range(6):
    outerX, outerY = getPointInDir(200, 200, 80+i*36, 192)
    innerX, innnerY = getPointInDir(200, 200, 80+i*36, 160)
    Line(outerX, outerY,innerX, innnerY,fill=rgb(225, 225, 0),lineWidth=9)
# Draw tick marks for fuel meter
for i in range(6):
    if (i % 4 != 0):
        outerX, outerY = getPointInDir(200, 55, 100+i*20, 63)
        innerX, innnerY = getPointInDir(200, 55, 100+i*20, 77)
        Line(outerX, outerY,innerX, innnerY,fill=rgb(225, 225, 0),lineWidth=5)
# Draw Numbers
Label('1',335,171,bold=True,fill=rgb(225, 225, 0),size=50)
Label('2',324,257,bold=True,fill=rgb(225, 225, 0),size=50)
Label('3',268,318,bold=True,fill=rgb(225, 225, 0),size=50)
Label('4',180,335,bold=True,fill=rgb(225, 225, 0),size=50)
Label('5',103,297,bold=True,fill=rgb(225, 225, 0),size=50)
Label('6',60,220,bold=True,fill=rgb(225, 225, 0),size=50)
Label('G/h',200,79,bold=True,fill=rgb(225, 225, 0),size=14)
Label('MPG',200,93,bold=True,fill=rgb(225, 225, 0),size=19)
Label('1',179,106,bold=True,fill=rgb(225, 225, 0),size=12)
Label('0.5',232,99,bold=True,fill=rgb(225, 225, 0),size=12)
Label('20',169,137,bold=True,fill=rgb(225, 225, 0),size=18)
Label('30',226,137,bold=True,fill=rgb(225, 225, 0),size=18)
Label('40',252,124,bold=True,fill=rgb(225, 225, 0),size=18)
Label('60',270,104,bold=True,fill=rgb(225, 225, 0),size=18)
# Draw 'redline'
Arc(200, 200, 400, 400, 273, 35, fill=rgb(225, 225, 0))
Arc(200, 200, 325, 325, 270, 41, fill=rgb(25, 25, 25))
# Draw screw dots
Circle(150,200,8,fill=rgb(45, 45, 45))
Circle(250,200,8,fill=rgb(45, 45, 45))
# Draw outside grey circle
Circle(200,200,202,fill=None,border=gradient('dimGray','black',start='right'),
borderWidth=12)
Circle(200,200,200,fill=None,border=gradient('lightGrey','dimGray',
start='right'),borderWidth=8)
# End of drawing Techometer components

# Draw the pointer and an invisible pointer opposite so it can rotate -
# - concentrically
pointer = Group(
Rect(194,200,12,180,fill=None),
Rect(194,30,12,170,fill=gradient(rgb(235, 195, 0),rgb(255, 255, 0),
rgb(235, 195, 0),start='left')),
RegularPolygon(200,27,7,3,fill=gradient(rgb(235, 195, 0),rgb(255, 255, 0),
rgb(235, 195, 0),start='left'))
)
# Draw pointer cover
Circle(200,200,27,fill=rgb(45, 45, 45))
# Draw a second pointer
```

```
pointer0 = Group(
Rect(194,200,12,60,fill=None),
Rect(194,30,12,170,fill=gradient(rgb(235, 195, 0),rgb(255, 255, 0),
rgb(235, 195, 0),start='left')),
RegularPolygon(200,27,7,3,fill=gradient(rgb(235, 195, 0),rgb(255, 255, 0),
rgb(235, 195, 0),start='left'))
)
# Adjust height, and width then position of second pointer
pointer0.width = pointer0.width * .7
pointer0.height = pointer0.height * .6
pointer0.centerY = 55
pointer0.rotateAngle = 90

# Draw 'reflections' as overlay
Circle(200,200,190,fill=gradient('white','black'),opacity=17)
Circle(200,200,190,fill=gradient('black','black','black','white','black',
'black','black',start='left'),opacity=7,rotateAngle=-35)

# Create some globals to keep track of statuses between onKeyHold, -
# - onKeyRelease, and onStep functions

# Keeps track of weather or not the 'engine has been started and idling'
app.running = False
# Communicates from onKeyHold to conditionals in onStep
app.enterHeld = False
# Keeps track of 'engine Rotations Per Second' to use as a variable to rotate -
# - the pointer
app.rPM = 0
# Communicates from onKeyHold to conditionals in onStep
app.accel = False
# Hekper for making the pointer wobble once in the 'redline'
app.wobbleHelper = False

# Make the app rotate the needle while enter or space is pressed
def onKeyHold(keys):
    # Enter is used to 'start the engine', and once started will not work
    if ('enter' in keys) and (app.running == False) and (app.rPM < 35):
        app.rPM += 2
    # if the engine is started space increases the RPMs when held
    if ('space' in keys) and (app.running == True):
        app.rPM += 3
        app.accel = True
# part of a helper to determine weather the needle is increasing
def onKeyRelease(keys):
    if ('space' in keys):
        app.accel = False
# responds to held keys and increases the needle, or decreases needle
def onStep():
    # if the enter key is held past ~900 RPM the engine 'starts'
    if (app.rPM > 35):
        app.running = True
    # if the engine does not reach idle when enter is pressed if falls back to 0
    if (app.rPM > 0) and (app.running == False):
        app.rPM -= 1
    # Make the needle rotation match the rPM variable (Which increases when -
    # - space or enter is held)
    pointer.rotateAngle = app.rPM+44
    if (app.rPM < 250):
        pointer0.rotateAngle = app.rPM*.57+105
    if (app.rPM > 25) and (app.running == True):
        app.rPM -= 1.5
# This following function makes the needle bounce back and forth while -
# - it is in the 'redline' and space is held
    # Establishes where the 'redline' starts in reference to angle
    if (app.rPM > 225) and (app.accel == True):
        app.rPM -= 1.5
        # These make it increase for a short bounded period, before -
        # - decreasing for a short bounded period
        if (app.rPM > 257) and (app.wobbleHelper == True):
            app.wobbleHelper = False
        elif (app.rPM < 230) and (app.wobbleHelper == False):
            app.wobbleHelper = True
        # Some helpers to randomize the up and down movement
        if (app.wobbleHelper == True):
            app.rPM += (randrange(1,36)/5)
        else:
```

```
150             app.rPM -= (randrange(1,36)/5)
```