Outline

1 The Widget Scale
   - making colors with scale widgets
   - entering parameter values
   - using the variables in Scale

2 Building an Animation
   - sliding canvas coordinates
   - animating a random walk
entering data with scale
developing animations

1. The Widget Scale
   - making colors with scale widgets
   - entering parameter values
   - using the variables in Scale

2. Building an Animation
   - sliding canvas coordinates
   - animating a random walk
an application: mixing red, green, blue
The variable set by the scale is a double, ranging from 0 to 1 with resolution \(1.0/256\).

After every change in the variable, the method `show_colors` is executed.

When the GUI starts up, the scale has value 0.5.

def __init__(self, wdw):
    ...
    self.red = DoubleVar()  # red intensity
    self.scared = Scale(wdw, orient='vertical',
                        length=self.dim, \
                        from_=0.0, to=1.0, resolution=1.0/256, \
                        variable=self.red, command=self.show_colors)
    self.scared.set(0.5)  # initial value of red scale
entering data with scale
developing animations

1. The Widget Scale
   - making colors with scale widgets
   - entering parameter values
   - using the variables in Scale

2. Building an Animation
   - sliding canvas coordinates
   - animating a random walk
the widget Scale

To enter parameter variables:

1. programmer can specify a meaningful range;
2. the scale is initialized with a default value.
entering data with scale
developing animations

1 The Widget Scale
- making colors with scale widgets
- entering parameter values
- using the variables in Scale

2 Building an Animation
- sliding canvas coordinates
- animating a random walk
using the variables in Scale

Observe:

1. the Entry widget will display the Scale variable;
2. only when the Button show value is pressed.
object oriented version in `exscale1.py`

- data attributes define the layout of the GUI,
- functional attribute define the actions.

```python
class ShowScale(object):
    """
    GUI to demonstrate use of a scale.
    """
    def __init__(self, wdw):
        """
        determines the layout of the GUI
        """
    def show_value(self):
        """
        Shows the value of the scale variable
        in the entry widget.
        """

The `show_value` is the callback function for Button.
```
adding command to Scale
avoiding Button

The Scale widget has the command option.

- Use as command = ShowValue.
- ShowValue is called whenever the user activates the Scale.
- ShowValue displays the value of the scale variable in the Entry widget.
entering data with scale
developing animations

1 The Widget Scale
- making colors with scale widgets
- entering parameter values
- using the variables in Scale

2 Building an Animation
- sliding canvas coordinates
- animating a random walk
Sliding a Dot
to feel the coordinates
The design of the GUI consists of:

- A horizontal scale for the x coordinate;
- A vertical scale for the y coordinate;
- A canvas to draw the dot and its coordinates.

Other data attributes are the values for $x$ and $y$.

The action performed by the GUI is to draw the dot on the canvas using the coordinates.

The action is triggered when the user touches the scales, adjusting the coordinates for the dot.
def main():
    """
    Instantiates the GUI and launches the main event loop.
    """

class SlideDot(object):
    """
    GUI to demonstrate canvas coordinates.
    """
    def __init__(self, wdw, size):
        """
        determines the layout of the GUI
        """
    def draw_dot(self, val):
        """
        draws the dot and its scale variable
        """
entering data with scale
developing animations

1. The Widget Scale
   - making colors with scale widgets
   - entering parameter values
   - using the variables in Scale

2. Building an Animation
   - sliding canvas coordinates
   - animating a random walk
A Moving Dot

animating a random walk
The layout of the animated random walk consists of
- the canvas on which the dot will move;
- a horizontal scale to regulate the speed;
- a vertical scale for the step size;
- start and stop buttons.

The actions of the GUI can be summarized as
- the animation starts when start is pressed;
- and stops when stop is pressed;
- any changes in the values set by the scale take immediate effect when the animation is running.
class MovingDot(object):
    """
    GUI to illustrate an animation.
    """
    def __init__(self, wdw, size):
        """
        determines the layout of the GUI
        """
    def animate(self):
        """
        performs the animation
        """
    def start(self):
        """
        starts the animation
        """
    def stop(self):
        """
        stops the animation
        """
Exercises

1. Define a scale to determine the formatting of floats. If \( k \) is the scale variable, it is used as ‘\%.k’.
   Use two Entry fields in the GUI: one for the number entered by the user and the other for the formatted number.

2. Use the scale in the GUI to evaluate expressions, adjusting `guievaloo.py` of Lecture 31.

3. Make an animation of a dot moving along a circle, centered at the center of the canvas.
   Use scales to adjust the speed of the animation and the radius of the circle.

4. Add a scale to `sliding_puzzle.py` of Lecture 30 to adjust the speed of the scrambling & unscrambling.