

# Tkinter animation

Marquette University

# Animation

- Animation consists of displaying frames
  - A frame is a still picture
  - Ultimately, computer redraws the viewing surface completely
- To make animation efficient,
  - Avoid redrawing the same elements
  - Learn how to predict and show movement

# Animation

- Modern computer graphics puts most of the graphical computation into the graphics card
- Computer graphics cards are so powerful that they can be used as processing engines

# Animation with Tkinter Canvas

- Tkinter canvas allows you to move objects
  - Need to retain the object id returned with the create\_method
- Sometimes simpler:
  - Redraw the canvas completely
  - Has bad performance but is quite simple

# Animation with Tkinter


- To present smooth interface:
  - Need to redraw canvas at least 25 times per second
  - Often requires too much calculation in the CPU

# Animation with Tkinter

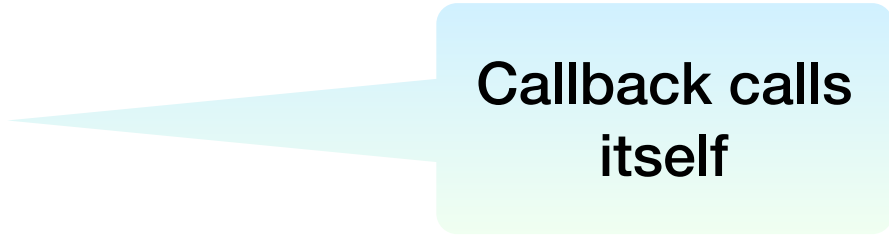
- Possibility 1:
  - Use the after method in order to call a callback function over and over again
    - Need to repeat the call within the callback function because otherwise it will only be called once

# Animation with Tkinter

```
class Test:
    def __init__(self):
        self.window = tk.Tk()
        self.window.after(2, self.alert)
        self.window.mainloop()
    def alert(self):
        print("hello")
        self.window.after(2, self.alert)
```



**Creating the  
callback**



**Callback calls  
itself**

# Animation with Tkinter

- Possibility 2:
  - Use an infinite loop
    - Still needs an initial “after” in order to be started
  - Use `time.sleep` in order to suspend animation long enough



# Animation with Tkinter

```
class Test:
    def __init__(self):
        self.window = tk.Tk()
        self.window.title("testing")
        self.window.after(2, self.alert)
        self.window.mainloop()
    def alert(self):
        while True:
            time.sleep(2)
            print("hello")
```

Creating the  
callback

Infinite loop

Using sleep to  
pause