Universidade Federal do Rio Grande
Centro de Ciências Computacionais

# Algoritmos e Estruturas de Dados I

# graphics.py

Profs. Drs. Cleo Billa, Rafael Penna e Thiago da Silveira

1° Semestre de 2020

# Roteiro

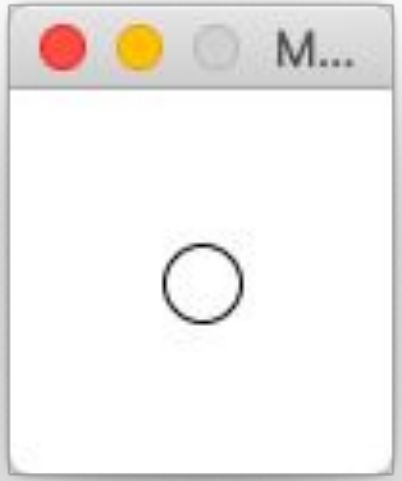
- Motivação
- Criar uma janela
- Desenhar na janela
- Imagens
- Eventos
- Mover elementos na janela

# Motivação

- Criar uma janela e desenhar nela
- Criar animações
- Criar jogos com gráficos
- Criar interfaces gráficas em python

# Desenhar uma janela

```
from graphics import *

def main():
    win = GraphWin("Minha Janela", 100, 100)
    c = Circle(Point(50,50), 10)
    c.draw(win)
    win.getMouse()  # Espera o click do mouse
    win.close()     # Fecha a janela

if __name__ == "__main__":
    main()
```



# Desenhando

```
from graphics import *
def main():
    win = GraphWin("Meu Campo", 200, 140)
    campo = Rectangle(Point(10,10),Point(190,130))
    campo.draw(win)
    circulo = Circle(Point(100,70), 25)
    circulo.draw(win)
    meiocampo = Line(Point(100,10),Point(100,130))
    meiocampo.draw(win)

    win.getMouse() # Espera o click do mouse
    win.close()    # Fecha a janela

if __name__ == "__main__":
    main()
```

# GraphWin

- GraphWin(title, width, height)
  - Cria um objeto do tipo janela com o título, largura e altura.
- Métodos:
  - setBackground(color):  determina a cor de fundo.
  - close(): fecha a janela.
- Exemplos:
  - Win = GraphWin("Minha Janela", 200,300)
  - Janela = GraphWin("Outra Janela", 300,200)
  - Win.close()
  - Janela.setBackground('blue')

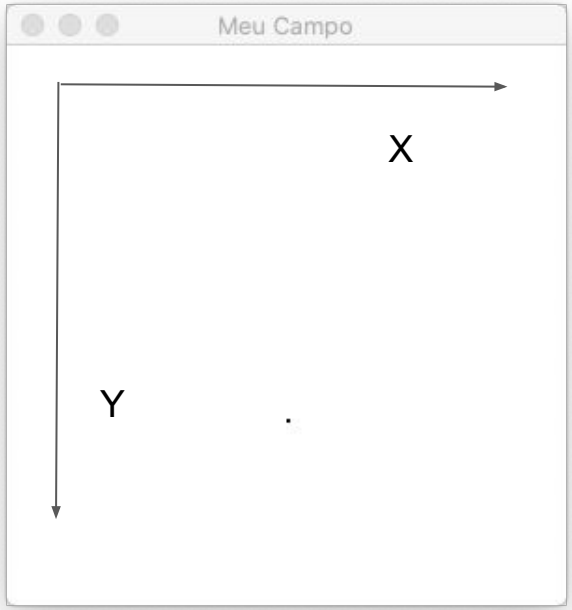
# Elementos Básicos de Desenho e Seus Principais Métodos

- Point
- Line
- Circle
- Oval
- Rectangle
- Polygon
- Text

- **draw(GraphWin)** → desenha o objeto na janela.
- **undraw()** → apaga objeto da janela.
- **setFill(color)** → determina a cor de preenchimento do objeto.
- **setOutline(color)** → configura a cor de contorno do objeto.
- **setWidth(pixels)** → determina a espessura da linha do objeto.
- **move(dx,dy)** → move o objeto dx unidades na direção x e dy unidades na direção y.

# Point

- Point(x,y)
  - Cria um ponto na coordenada x,y
- Métodos (além dos mencionados):
  - getX() retorna a coordenada x do ponto.
  - getY() retorna a coordenada y do ponto.
- Exemplos
  - P = Point(150, 200) # Cria um ponto onde x=150 e y=200
  - P.draw(win)
  - X = P.getX() # retorna x (150)



# Line

- Line(point1, point2)
  - Cria uma linha do ponto 1 ao ponto 2.
- Métodos (além dos mencionados):
  - getCenter(): retorna um clone do ponto central da linha.
  - getP1() retorna um clone do ponto 1 da linha
  - getP2() retorna um clone do ponto 2 da linha
- Exemplos:
  - L = Line(Point(10,20),Point(30,40))
  - L.draw(win)
  - P1 = L.getP1() # retorna o ponto Point(10,20)

# Circle

- Circle(centerPoint, radius)
  - Cria um círculo no ponto determinado e com o raio determinado.
- Métodos (além dos mencionados):
  - getCenter(): retorna um clone do ponto central do círculo.
  - getRadius(): retorna o raio do círculo.
- Exemplos:
  - C = Circle(Point(10,20),15)
  - Centro = C.getCenter() # retorna o ponto (10,20)
  - Raio = C.getRadius() # retorna o raio (15)

# Text

- Text(anchorPoint, string)
  - Cria um objeto do tipo texto no ponto determinado.
- Métodos (além de draw() e undraw())
  - setText(string):  Define o texto do objeto.
  - getText(): retorna a string do objeto.
  - getAnchor(): retorna um clone do ponto central do objeto.
  - setFace(family): define a fonte (Disponíveis: ’helvetica’, ’courier’, ’times, roman’ e ’arial’).
  - setSize(size): define o tamanho da fonte (5-36).
  - setStyle(style): define o estilo da fonte (Disponíveis: ’normal’, ’bold’, ’italic’ e ’bold italic’).
  - setTextColor(color): define a cor do texto.
- Exemplos:
  - Texto = Text(Point(10,10),"Exemplo")
  - Texto.setSize(20)
  - Texto.draw(win)

# Entry

- Entry(centerPoint, width)
  - Cria um objeto do tipo Entry (entrada de texto) no ponto determinado.
- Métodos além de (draw() e undraw()):
  - setText(string):  Define o texto do objeto.
  - getText(): retorna a string do objeto.
  - getAnchor(): retorna um clone do ponto central do objeto.
  - setFace(family): muda a fonte (Disponíveis: ’helvetica’, ’courier’, ’times, roman’ e ’arial’).
  - setSize(size): muda o tamanho da fonte (5-36).
  - setStyle(style) muda o estilo da fonte (Disponíveis: ’normal’, ’bold’, ’italic’ e ’bold italic’).
  - setTextColor(color) muda a cor do texto.
- Exemplos:
  - entrada = Entry(Point(10,10),30)
  - entrada.draw(win)
  - Texto = entrada.getText()

# Image

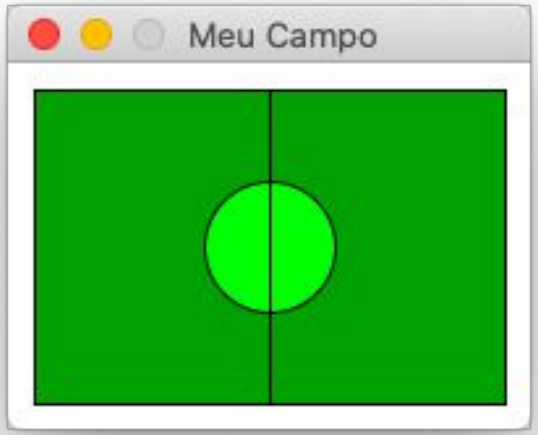
- Image(anchorPoint, fileName) ou
- Image(anchorPoint, width, height)
  - Cria um objeto do tipo imagem centralizado no ponto determinado. Pode ser baseado em um arquivo ou em branco.
  - Abre apenas imagens .gif
- Métodos além de (draw() e undraw()):
  - getAnchor(): retorna um clone do ponto central do objeto.
  - getWidth(): retorna a largura da imagem em pixels.
  - getHeight(): retorna a altura da imagem em pixels.
  - getPixel(x,y): retorna uma lista [red, green, blue] dos valores RGB do ponto (x,y).
  - setPixel(x,y,color): pinta o ponto (x,y) da imagem com uma cor.
  - save(fileName): grava uma imagem.
- Exemplos:
  - img = Image(Point(100,100),"bola.gif")
  - img.draw(win)

# Cores

- As cores mais tradicionais podem ser indicadas usando strings:
  - "blue","red","yellow","green","purple","cyan",...
- Cores mais tradicionais podem ter variações também:
  - "red1","red2","red3",...
- Para a lista completa veja: https://www.w3schools.com/colors/colors_x11.asp
- A graphics.py também permite criar suas próprias cores usando a função color_rgb(red, green, blue).
  - Exemplo:
    - color_rgb(255, 0, 0) é vermelho
    - color_rgb(0, 0, 0) é preto
    - color_rgb(255, 255, 255) é branco

# Exemplo

```
from graphics import *
def main():
    win = GraphWin("Meu Campo", 200, 140)
    campo = Rectangle(Point(10,10),Point(190,130))
    campo.setFill("green4")
    campo.draw(win)
    circulo = Circle(Point(100,70), 25)
    circulo.setFill("green1")
    circulo.draw(win)
    meiocampo=Line(Point(100,10),Point(100,130))
    meiocampo.draw(win)

    win.getMouse()  # Espera o click do mouse
    win.close()     # Fecha a janela
if __name__ == "__main__":
    main()
```

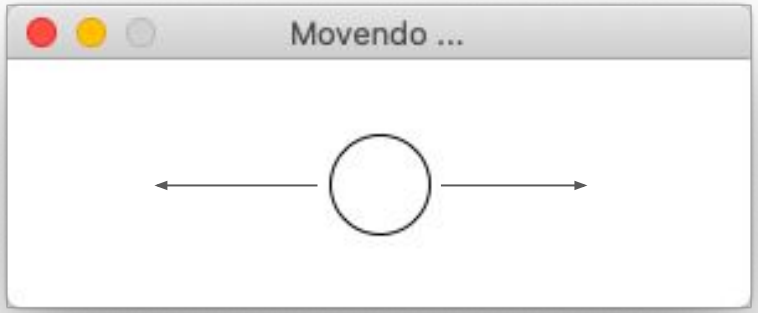


# Mais de GraphWin

- GraphWin(title, width, height)
  - Cria um objeto do tipo janela com o título, largura e altura.
- Métodos:
  - setBackground(color):  determina a cor de fundo.
  - close(): fecha a janela.
  - **getMouse():** pausa o programa até o usuário clicar na janela. Retorna o ponto onde o mouse foi clicado.
  - **checkMouse():** similar ao getMouse() mas não para o programa.
  - **getKey():** pausa o programa até o usuário pressionar uma tecla. A janela deve estar ativa quando o usuário aperta a tecla. Retorna a tecla pressionada ("Left", "Right", "Up", "a", "A", "space", …)
  - **checkKey():** Similar ao getKey() mas não pausa o programa.

# move(dx,dy)

```
from graphics import *
def main():
    win = GraphWin("Movendo ...", 300, 100)
    c=Circle(Point(150,50),20)
    c.draw(win)
    key=win.checkKey()
    while key!="Escape":
        if key == "Right":
            c.move(3,0)
        elif key == "Left":
            c.move(-3,0)
        key=win.checkKey()

if __name__ == "__main__":
    main()
```

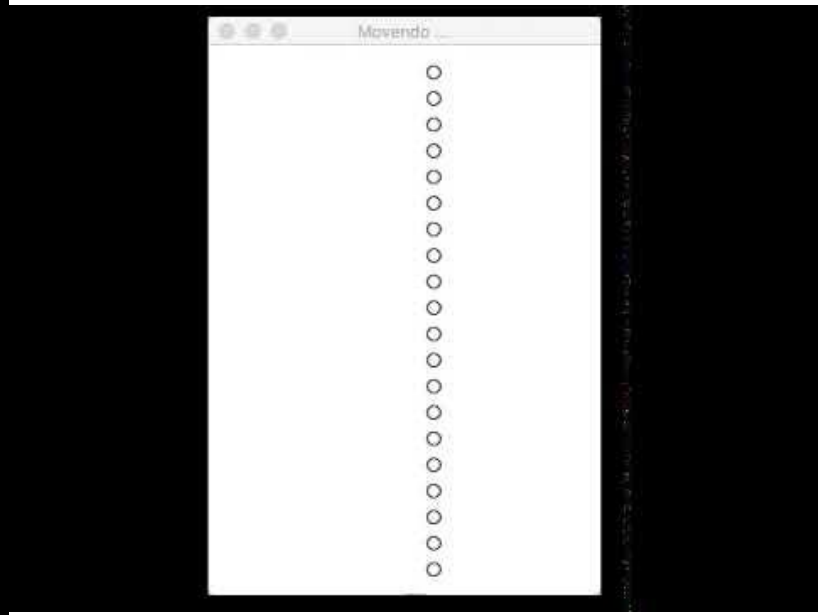


# Animação: autoflush=True (padrão)

```
from graphics import *
def main():
  win = GraphWin("Movendo ...",300,420)
  lC=[]
  for i in range(0,20):
      c=Circle(Point(0,(i+1)*20),5)
      c.draw(win)
      lC.append(c)

  key=win.checkKey()
  while key!="Escape":
     for c in lC:
          c.move(1,0)
     key=win.checkKey()

if __name__ == "__main__":
  main()
```

# Animação: autoflush=False

```
from graphics import *
def main():
  win = GraphWin("Movendo ...",300,420,autoflush=False)
  lC=[]
  for i in range(0,20):
      c=Circle(Point(0,(i+1)*20),5)
      c.draw(win)
      lC.append(c)

  key=win.checkKey()
  while key!="Escape":
     for c in lC:
          c.move(1,0)
     key=win.checkKey()
     update(30)

if __name__ == "__main__":
  main()
```



# Mais da graphics.py

- graphics.py: https://mcsp.wartburg.edu/zelle/python/graphics.py
- https://mcsp.wartburg.edu/zelle/python/
  - HTML: https://mcsp.wartburg.edu/zelle/python/graphics/graphics/
  - PDF: https://mcsp.wartburg.edu/zelle/python/graphics/graphics.pdf
  - Chapter04.pptx
- http://anh.cs.luc.edu/handsonPythonTutorial/graphics.html
- Python Graphics Programming: Graphics.py
- https://pypi.org/project/graphics.py/

Universidade Federal do Rio Grande
Centro de Ciências Computacionais

# Algoritmos e Estruturas de Dados I

# graphics.py

Profs. Drs. Cleo Billa, Rafael Penna e Thiago da Silveira

1° Semestre de 2020