turtle: manual de referência (muito) rápida

Ernesto Costa
Departamento de Engenharia Informática
Universidade de Coimbra
Versão 0.2

30 de Setembro de 2012

Resumo

Este texto é um pequeno e breve manual do utilizador para o módulo **turtle**. Indica-se no texto o local onde o código do módulo pode ser obtido.O módulo vem conjuntamente com a distribuição de **Python** pelo que não precisa fazer nada o instalar [1]. O texto apresenta os diversos comandos agrupados em secções, sendo nos casos que justifique, acompanhado por exemplo ilustrativos. O texto está em construção e por isso está ainda incompleto. A consulta da documetação de **Python** ajudará a completar esta informação.

1 Introdução

A linguagem **Python** tem um conjunto de módulos adicionais que permitem aumentar muito o seu poder expressivo. Existe um que permite efectuar desenhos, Baseia-se na ideia de uma **tartaruga** que se passeia sob nosso controlo, podendo deixar um rasto visível. Para isso tem uma caneta. Sempre que a caneta está em baixo e a tartaruga caminha, lá aparece o rasto. Existem por isso comandos para baixar a caneta, levantar a caneta, rodar para a esquerda ou para a direita, etc. Quando instala o Python fica com a possibilidade de importar de imediato o módulo **turtle**. Outras possibilidades são o **xturtle** ou a sua versão mais completa **turtle** (http://www.cs.luther.edu/~pythonworks/PythonContext/). Iremos utilizar neste documento a versão que vem com a distribuição da linguagem.

[1] No entanto, caso use alguma das variantes disponíveis (**cTurtle**, **xTurtle**), terá que tomar algumas precauções na colocação desses módulos. Deverá ser colocado num local onde o **Python** o posso descobrir, tipicamente na pasta **site-packages**.

Comecemos com um exemplo simples que nos permite desenhar um quadrado, com um determinado lado. Para além disso controlamos o local onde é desenhado e a sua orientação. O código aparece na listagem 1.

```
1  import turtle
2
3  def quadrado(lado,pos, angulo):
4      """
5      Desenha um quadrado com o comprimento de lado, o vértice
           inferior
6      esquerdo em pos e  direcção inicial angulo.
7      """
8      # Preparação
9      turtle.up()
10     turtle.goto(pos)
11     turtle.setheading(angulo)
12     turtle.down()
13
14     # desenha quadrado
15     for conta in range(4):
16         turtle.forward(lado)
17         turtle.right(90)
18     turtle.hideturtle()
```

Listagem 1: Sai um quadrado ...

Analisemos o seu funcionamento. Numa primeira parte, preparamos o programa para uma correcta execução: levantamos a caneta (linha 9), vamos para a posição que nos é indicada na forma **x,y** (linha 10), orientamos a caneta (linha 11) e colocamo-la em baixo para desenhar (linha 12). Depois é trivial. Repetimos quatro vezes desenhar um lado (linha 16) e rodar 90 graus para a direita (linha 17) . A figura 1 mostra o exemplo de uma execução quando o lado é 50 a posição (-50,50) e o ângulo 35 graus.

Neste exemplo usámos um conjunto de **funções** disponibilizadas pelo módulo **turtle**. Usando a instrução **import turtle**, somos obrigados a préfixar o nomes dessas operações com o nome do módulo quando se trata do seu **uso**. O mesmo resultado pode ser obtido recorrendo a objectos do tipo **Turtle**, criados explicitamente por nós. Neste caso passamos a usar os **métodos** associados aos objectos desse tipo. Os nomes (e funcionalidades) são basicamente os mesmos. A listagem 2 mostra o código.

```
1  import turtle
```

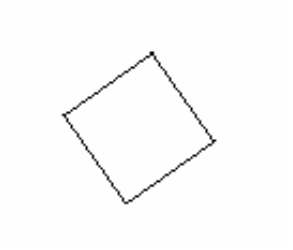

Figura 1: **Quadrado**

```
2
3  def quadrado(tartaruga,lado,pos,angulo):
4      """ Usa a tartaruga para desenhar um quadrado de lado
5      lado e orientação inicial ângulo.
6      """
7      # Preparação
8      tartaruga.up()
9      tartaruga.goto(pos)
10     tartaruga.setheading(angulo)
11     tartaruga.down()
12
13     # desenha quadrado
14     for conta in range(4):
15         tartaruga.forward(lado)
16         tartaruga.right(90)
17     tartaruga.hideturtle()
18
19 def main0():
20     tarta = turtle.Turtle()
21     quadrado(tarta,100,(50,50),45)
22     turtle.exitonclick()
```

Listagem 2: Saia um quadrado ...

Se os métodos e as funções fazem o mesmo, então qual a vantagem de ter ambos? A resposta é simples: basta imaginar que se tem mais do que uma tartaruga que quer comandar de modo autónomo! A classe **Turtle** permite isso ao poder originar tantos objectos, i.e., tartarugas, quantos queiramos. Vejamos (ver listagem 3) um exemplo simples, agora com circunferências.

```
1  import turtle
2
```

```
3  def circunferencia(tartaruga,raio, posicao):
4      # Preparação
5      tartaruga.up()
6      tartaruga.goto(posicao)
7      tartaruga.down()
8
9      # desenha
10     tartaruga.circle(raio)
11
12     tartaruga.hideturtle()
13
14 def main():
15     tarta1 = turtle.Turtle()
16     tarta1.begin_fill()
17     tarta1.pen(shown=True,pendown=False,pencolor='blue',
           fillcolor='blue', speed=5)
18     circunferencia(tarta1,30,(50,50))
19     tarta1.end_fill()
20
21     tarta2 = turtle.Turtle()
22     tarta2.begin_fill()
23     tarta2.pen(shown=False,pendown=False,pencolor='red',
           fillcolor='red', speed=2)
24     circunferencia(tarta2, 100, (-100,-100))
25     tarta2.end_fill()
26     turtle.exitonclick()
```

Listagem 3: Saiam dois quadrados ...

E a respectiva imagem, que se mostra na figura 2.

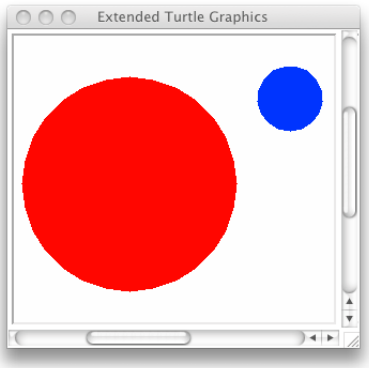

Figura 2: **Duas Circunferências**

2 Acção: comandos básicos

O elementar

Os movimentos básicos podem ser feitos em função da direcção actual da tartaruga, ou em função do ponto de chegada.

Nome	Descrição
forward(unidades) ou fd(unidades)	Avança na direcção em que se encontra tantas unidades
back(unidades) ou bk(unidades)	Recua na direcção em que se encontra tantas unidades
goto(pos,y=None)	Movimenta a tartaruga para a posição indicada
setx(x)	Movimenta a tartaruga para a posição de coordenada x. y mantém-se.
sety(y)	Movimenta a tartaruga para a posição de coordenada y. x mantém-se.

Tabela 1: Movimentos

forward

argumento	um número (inteiro ou vírgula flutuante)
chamada	forward(número) ou fd(número)

Exemplo	`turtle.forward(50)`
	`turtle.goto((50,50))`
	`turtle.back(25)`
	`turtle.goto(25,75)`
	`turtle.setx(75)`
	`turtle.sety(25)`

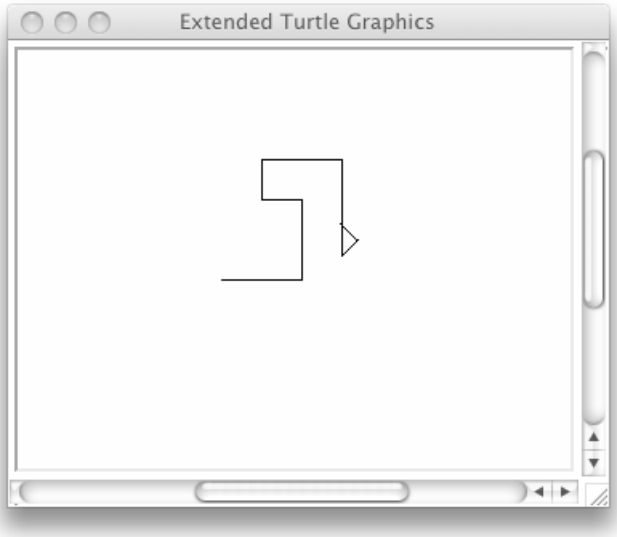

Figura 3: **Movimentos básicos**

Controlar a orientação

Nome	Descrição
right(angulo)	Roda para a direita o ângulo.
left(ângulo)	Roda para a esquerda o ângulo.
setheading(ângulo)	Orientação absoluta

Tabela 2: Rotações

right

argumento	um número (inteiro ou vírgula flutuante)
chamada	right(número) ou rt(número)

Exemplo	`>>> import turtle`
	`>>> turtle.heading()`
	`0.0`
	`>>> turtle.right(45)`
	`>>> turtle.heading()`
	`315.0`
	`>>>`

Alguns desenhos e escritas

Nome	Descrição
circle(raio,amplitude)	Desenha um círculo ou um arco de círculo.
dot(tamanho=None, *cor)	Desenha um ponto na posição em que se encontra.
write(texto, move=False, align='left', font =('Arial',8,'normal'))	Escreve texto na posição corrente.Não se move se False.

Tabela 3: Pequenas coisas.

circle

argumentos	número
argumentos opcionais	número, inteiro
chamada	circle(número) # círculo completo
	circle(número, número) # arco
	circle(número, número, inteiro)

Exemplo	`>>> turtle.circle(50)`
	`>>> turtle.circle(120,180) # semi-círculo`
	`>>>`

3 Estados

O estado da tartaruga

Nome	Descrição
position()	Devolve a posição actual da tartaruga na forma de um tuplo (x,y).
towards(pos,y=None)	Devolve o ângulo entre a orientação actual da tartaruga e a posição passada como argumento.
xcor()	Devolve a posição da tartaruga relativa ao eixo dos x.
ycor()	Devolve a posição da tartaruga relativa ao eixo dos y.
heading()	Devolve a orientação actual da tartaruga.
distance(pos,y=None)	Devolve a distância entre a posição actual da tartaruga e a posição passada como parâmetro.
hideturtle()	Esconde a tartaruga.
showturtle()	Mostra a tartaruga.
shape(nome=None)	Define a forma da tartaruga.

Tabela 4: Estado da tartaruga

O estado da caneta

Nome	Descrição
down() ou pd()	Coloca a caneta em baixo.
up() ou pu()	Coloca a caneta no ar.
pen()	Devolve ou define os atributos da caneta.

Tabela 5: Estado da caneta

pen

argumentos	diversos tipos
argumentos opcionais	número, inteiro
chamada	pen() # consulta
	pen(fillcolor='black', pencolor='red',pensize=10) # define alguns atributos para todas as canetas
	tarta.color('yellow','') # Define atributo para uma caneta específica

Exemplo	`>>> turtle.pen(fillcolor="black", pencolor="red", pensize=10)`
	`>>> sorted(turtle.pen().items())`
	`[('fillcolor', 'black'), ('outline', 1), ('pencolor', 'red'),`
	`('pendown', True), ('pensize', 10), ('resizemode', 'noresize'),`
	`('shown', True), ('speed', 9), ('stretchfactor', (1, 1)), ('tilt', 0)]`
	`>>> penstate=turtle.pen()`
	`>>> turtle.color("yellow", "")`
	`>>> turtle.penup()`
	`>>> sorted(turtle.pen().items())`
	`[('fillcolor', ''), ('outline', 1), ('pencolor', 'yellow'),`
	`('pendown', False), ('pensize', 10), ('resizemode', 'noresize'),`
	`('shown', True), ('speed', 9), ('stretchfactor', (1, 1)), ('tilt', 0)]`
	`>>> turtle.pen(penstate, fillcolor="green")`
	`>>> sorted(turtle.pen().items())`
	`[('fillcolor', 'green'), ('outline', 1), ('pencolor', 'red'),`
	`('pendown', True), ('pensize', 10), ('resizemode', 'noresize'),`
	`('shown', True), ('speed', 9), ('stretchfactor', (1, 1)), ('tilt', 0)]`

4 Configurações diversas

Preenchimento

Nome	Descrição
begin_fill()	A chamar apara activar preenchimento.
end_fill()	A chamar para desactivar preenchimento.

Tabela 6: Preenchimento

begin_fill

argumento	não tem
chamada	—

Exemplo	`>>> turtle.begin_fill()`
	`>>> for i in range(4):`
	`...  turtle.forward(100)`
	`...  turtle.right(90)`
	`...`
	`>>> turtle.end_fill()`
	`>>>`

Medidas

Nome	Descrição
degrees(circulo = 360.0)	Altera para graus.
radians()	Altera para radianos.

Tabela 7: Medidas

Velocidade

Nome	Descrição
speed(velocidade=None)	Sem argumento, devolve. Com argumento, altera velocidade.
delay(atraso = None)	Sem nada devolve, com valor altera. Em milisegundos.

Tabela 8: Velocidade

Cor

Nome	Descrição
bgcolor(*args)	Altera ou devolve a cor de fundo.
bgpic(picname=None)	Coloca uma imagem como fundo da janela.
pencolor(*args)	Altera a cor da caneta.
fillcolor(*args)	Define a cor de preenchimento.
color(*args)	Sem parâmetros devolve as cores da caneta e de preenchimento.
colormode(cmodo=None)	Altera a especificação RGB para modo float ou inteiro.

Tabela 9: Cor

bgcolor

argumento	cadeia de caracteres ou 3-tuplo de valores 0.. colormode
chamada	bgcolor(cadeia caracteres) ou bgcolor((r,g,b))

Exemplo	`>>> turtle.bgcolor('magenta')`
	`>>> turtle.bgcolor()`
	`'magenta'`
	`>>> turtle.bgcolor(0.25,0,0.25)`
	`>>> turtle.bgcolor()`
	`'#400040'`
	`>>>`

5 Eventos

– TBD

6 Coisas diversas

Nome	Descrição
exitonclick()	Quando o rato clica dentro da janela fecha-a e abandona.
setworldcoordinates (llx,lly,urx,ury)	Altera o sistemas de coordenadas.

Tabela 10: Diversos