Anexo B - **Abstracção Janela gráfica**

Os procedimentos gráficos disponibilizados pelas várias implementações da linguagem Scheme são, em geral, sofisticados e exigem alguma experiência para uma utilização correcta. Para os exercícios e projectos propostos que requeiram interacção gráfica, optou-se pela definição e implementação de um conjunto de procedimentos simples que funcionam numa janela gráfica. Trata-se de uma abstracção, a *janela gráfica*, que esconde os pormenores dos procedimentos gráficos do *DrScheme*, dando ao programador a hipótese de controlar, de uma forma simples, uma caneta que pode desenhar, pintar, ou escrever texto, com cores à escolha. O ponto da janela onde se encontra a caneta é o chamado *ponto-corrente* e as suas coordenadas e as do cursor comandado pelo rato podem ser lidas, sempre que for necessário. Também está disponível a funcionalidade de leitura de tecla logo que actuada.

**Tabela de cores utilizada**

| índice | R | G | B | cor |
|---|---|---|---|---|
| 0 | 0.0 | 0.0 | 0.0 | preto |
| 1 | 0.0 | 0.0 | 0.5 | azul escuro |
| 2 | 0.0 | 0.0 | 1.0 | azul |
| 3 | 0.0 | 0.5 | 0.0 | verde escuro |
| 4 | 0.0 | 0.5 | 0.5 | cyan escuro |
| 5 | 0.0 | 0.5 | 1.0 | azul cyan |
| 6 | 0.0 | 1.0 | 0.0 | verde |
| 7 | 0.0 | 1.0 | 0.5 | verde cyan |
| 8 | 0.0 | 1.0 | 1.0 | cyan |
| 9 | 0.5 | 0.0 | 0.0 | vermelho escuro |
| 10 | 0.5 | 0.0 | 0.5 | magenta escuro |
| 11 | 0.5 | 0.0 | 1.0 | azul magenta |
| 12 | 0.5 | 0.5 | 0.0 | amarelo escuro |
| 13 | 0.5 | 0.5 | 0.5 | cinzento |
| 14 | 0.5 | 0.5 | 1.0 | azul cinza |
| 15 | 0.5 | 1.0 | 0.0 | verde amarelado |
| 16 | 0.5 | 1.0 | 0.5 | verde cinza |
| 17 | 0.5 | 1.0 | 1.0 | cyan pálido |
| 18 | 1.0 | 0.0 | 0.0 | vermelho |
| 19 | 1.0 | 0.0 | 0.5 | vermelho magenta |
| 20 | 1.0 | 0.0 | 1.0 | magenta |
| 21 | 1.0 | 0.5 | 0.0 | vermelho amarelado |
| 22 | 1.0 | 0.5 | 0.5 | vermelho cinza |
| 23 | 1.0 | 0.5 | 1.0 | magenta pálido |
| 24 | 1.0 | 1.0 | 0.0 | amarelo |
| 25 | 1.0 | 1.0 | 0.5 | amarelo pálido |
| 26 | 1.0 | 1.0 | 1.0 | branco |

- `(janela largura altura titulo)`

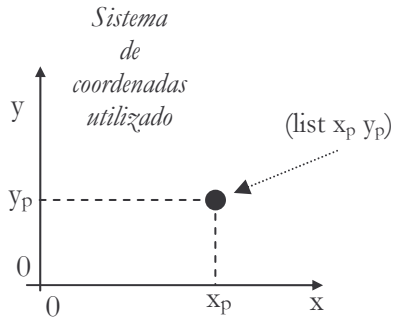
Cria janela gráfica com a dimensão *largura* x *altura* e com um título (cadeia de caracteres especificada por *titulo*).
Devolve uma lista com as características da janela, ou seja, `(largura altura titulo)`.

- `(cor indice-ou-rgb)`

De uma tabela de 27 cores, escolhe uma delas para cor da caneta, através de *indice-ou-rgb*, inteiro situado entre 0 e 26. Se *indice-ou-rgb* = *'temp* (temporário), o efeito de uma primeira passagem da caneta é anulado por uma segunda passagem. É com esta cor que se deve desenhar ou pintar um objecto animado, sem se perder os gráficos visualizados e sobre os quais o objecto se desloca. Para tal, bastará desenhar o objecto animado com a cor *'temp* e, quando ele vai sair, deverá ser novamente desenhado com a cor *'temp*, no local onde se encontra, reaparecendo o que ele tapava.
Ao parâmetro *indice-ou-rgb* também se podem associar três valores entre 0.0 e 1.0, que traduzem, respectivamente, os pesos R, G e B das cores Vermelho (Red), Verde (Green) e Azul (Blue). Por exemplo, com R=G=1.0 e B a variar de 0.0 a 1.0, codificam as cores desde o amarelo, passando pelo amarelo pálido até ao branco.

- `(move pt)`

Move a caneta, sem desenhar, para o ponto definido por *pt* (*pt* é uma lista com as coordenadas *x* e *y* do ponto).




- `(move-rel pt)`

Procede como *move*, mas o ponto *pt* é definido em relação ao *ponto corrente*. Sendo *Px* e *Py* as coordenadas do *ponto corrente*, e o argumento *pt* uma lista com *dx* e *dy*, a caneta é movida, sem desenhar, para o ponto definido pela lista (*Px* + *dx* *Py* + *dy*).

- `(le-cor ponto)`

Devolve uma lista com os coeficientes RGB da cor lida no pixel da janela referido por ponto.

- `(le-indice-cor ponto)`

Devolve índice (da tabela de cores) da cor lida no pixel da janela referido por ponto.
Devolve -1 se não está na tabela de cores.

- `(desenha sequencia-de-pontos)`

Sendo *sequencia-de-pontos* uma lista de pontos, desenha a linha poligonal definida pelos pontos em *sequencia-de-pontos*. A caneta fica sobre o último ponto da sequência.

- `(desenha-oval dois-pontos)`

Sendo *dois-pontos* uma lista de dois pontos, desenha a oval inscrita no rectângulo cujo vértice superior-esquerdo é o primeiro ponto da lista *dois-pontos* e o vértice inferior-direito é o segundo ponto. No final, a caneta retoma a posição que tinha à partida.

- `(pinta sequencia-de-pontos)`

Procede como *desenha*, mas preenche com a cor da caneta todo o interior do polígono definido por *sequencia-de-pontos*. Se o último ponto de *sequencia-de-pontos* não coincidir com o primeiro, o polígono é fechado ligando o último ponto com o primeiro, mas a caneta fica no último ponto.

- `(pinta-oval dois-pontos)`

Procede como *desenha-oval*, mas preenche com a cor da caneta o interior da oval.

- `(ponto pt)`

Procede como *move*, mas pinta com a caneta o ponto definido por *pt* e, no final, a caneta fica sobre esse ponto.

- `(desenha-rel sequencia-de-pontos)`

Procede como *desenha*, mas cada ponto em *sequencia-de-pontos* é definido relativamente ao ponto onde se encontra a caneta, antes desta começar a mover-se na sua direcção. No final, a caneta ficará sobre o último ponto da sequência.

- `(desenha-oval-rel dois-pontos)`

Procede como *desenha-oval*, mas os pontos da lista *dois-pontos* são definidos relativamente ao *ponto corrente*.

- `(pinta-rel sequencia-de-pontos)`

Procede como *pinta*, mas cada ponto da *sequencia-de-pontos* é definido relativamente ao ponto onde se

encontra a caneta, antes desta começar a mover-se na sua direcção.

- `(ponto-rel pt)`

Procede como *ponto*, mas *pt* é relativo ao *ponto corrente*.

- `(desenha-txt texto)`

A partir do *ponto corrente*, escreve a cadeia de caracteres *texto* e deixa a caneta no canto inferior direito do último carácter.

- `(caneta)`

Devolve a posição actual da caneta, ou seja, o *ponto corrente*.

- `(cursor)`

Devolve o ponto do interior da janela apontado pelo cursor, quando é actuado o botão esquerdo do rato.

- `(limpa)`

Limpa a janela.

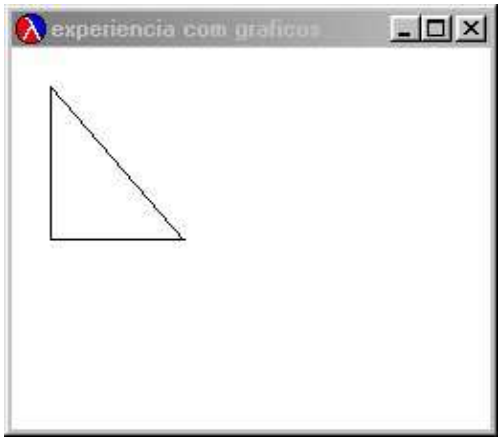
- `(tecla-pressionada)`

Devolve o código a tecla actuada.

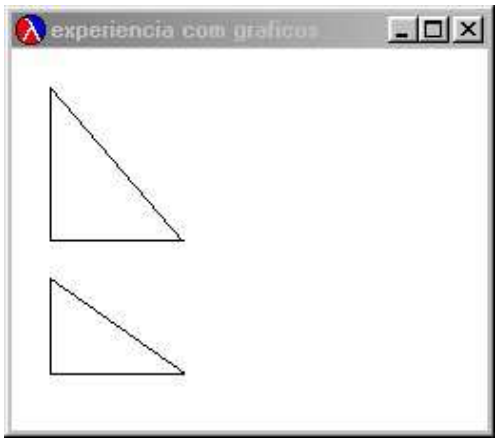
- `(fecha-janela)`

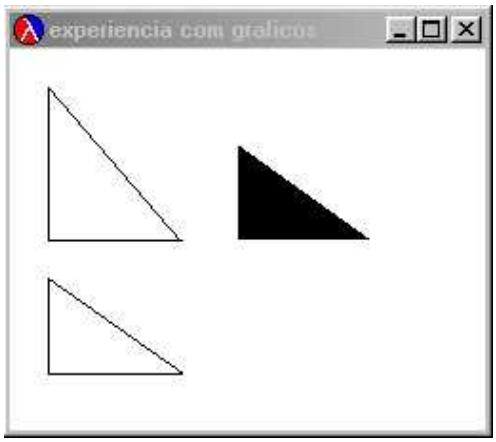
Fecha a janela gráfica.

```
> (require (lib "swgr.scm" "user-feup"))
> (janela 250 200 "experiencia com graficos")
(250 200 "experiencia com graficos")
> (desenha (list '(20 100)'(20 180)'(90 100)'(20 100)))
```

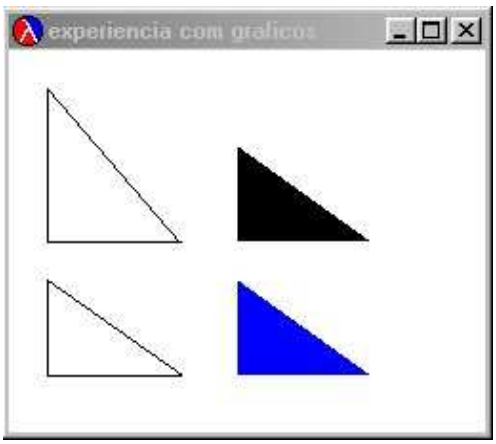



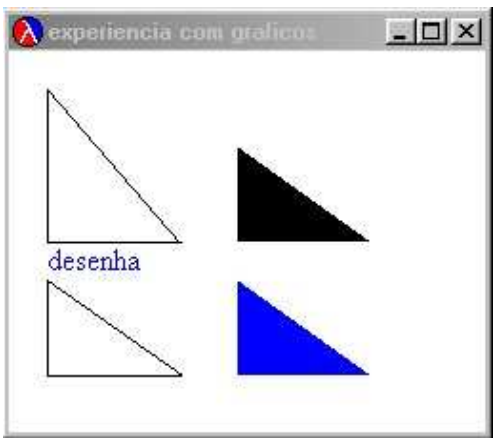

```
> (move (list 20 30))
> (desenha-rel (list '(0 0)'(0 50)'(70 -50)'(-70 0)))
> (move (list 120 100))
> (pinta-rel (list '(0 0)'(0 50)'(70 -50)'(-70 0)))
```



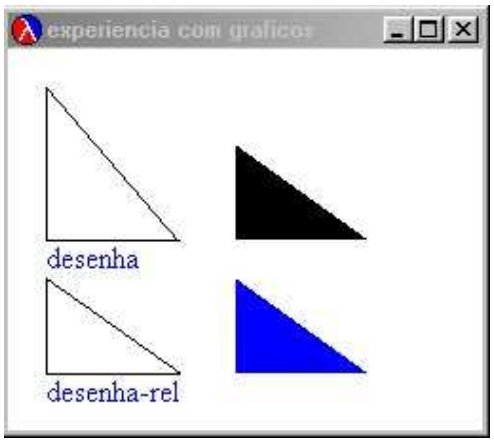



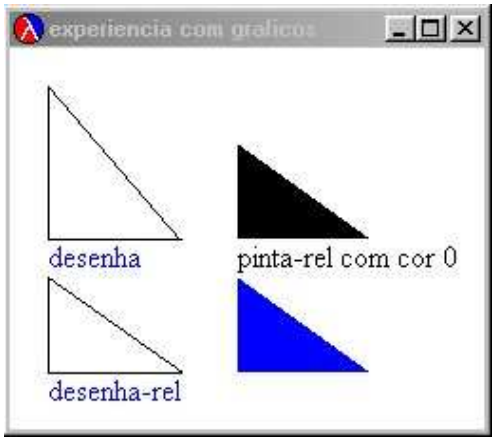

```
> (cor 2)
> (pinta '((120 30)(120 80)(190 30)))
> (move '(20 85))
> (desenha-txt "desenha")
```



```
> (move '(20 15))
> (desenha-txt "desenha-rel")
> (move '(120 85))
> (cor 0)
> (desenha-txt "pinta-rel com cor 0")
```




```
> (move (list 120 15))
> (cor 2)
> (desenha-txt "pinta com cor 2")
```