**[graphics.h](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics-h.html)**

Esta interface fornece acesso a uma biblioteca de gráficos simples que permite desenhar linhas, retângulos, ovais, arcos, polígonos, imagens e cordas em uma janela gráfica.

|  |
| --- |
| **Funções** |
| [**initGraphics ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:initGraphics)[**initGraphics (largura, altura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:initGraphics) | Cria a janela de gráficos na tela. |
| [**drawArc (limites, início, varredura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawArc)[**drawArc (x, y, largura, altura, início, varredura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawArc) | Desenha um arco elíptico inscrito em um retângulo. |
| [**fillArc (limites, início, varredura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillArc)[**fillArc (x, y, largura, altura, início, varredura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillArc) | Preenche uma área em forma de cunha de um arco elíptico. |
| [**drawImage (filename, pt)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawImage)[**drawImage (filename, x, y)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawImage)[**drawImage (filename, bounds)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawImage)[**drawImage (filename, x, y, width, height)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawImage) | Desenha a imagem do arquivo especificado com o canto superior esquerdo no ponto especificado. |
| [**getImageBounds (filename)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getImageBounds) | Retorna os limites da imagem contida no arquivo especificado. |
| [**drawLine (p0, p1)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawLine)[**drawLine (x0, y0, x1, y1)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawLine) | Desenha uma linha que liga os pontos especificados. |
| [**drawPolarLine (p0, r, theta)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawPolarLine)[**drawPolarLine (x0, y0, r, theta)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawPolarLine) | Desenha uma linha de comprimento **r**na direção **theta**do ponto inicial. |
| [**drawOval (limites)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawOval)[**drawOval (x, y, largura, altura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawOval) | Desenha a moldura de um oval com os limites especificados. |
| [**fillOval (limites)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillOval)[**fillOval (x, y, largura, altura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillOval) | Preenche o quadro de um oval com os limites especificados. |
| [**drawRect (limites)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawRect)[**drawRect (x, y, largura, altura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawRect) | Desenha a moldura de um retângulo com os limites especificados. |
| [**fillRect (limites)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillRect)[**fillRect (x, y, largura, altura)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillRect) | Preenche o quadro de um retângulo com os limites especificados. |
| [**drawPolygon (polygon)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawPolygon)[**drawPolygon (polygon, pt)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawPolygon)[**drawPolygon (polígono, x, y)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawPolygon) | Desenha o contorno do polígono especificado. |
| [**preencherPolygon (polígono)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillPolygon)[**preencherPolygon (polígono, pt)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillPolygon)[**preencherPolygon (polígono, x, y)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:fillPolygon) | Preenche o quadro do polígono especificado. |
| [**drawString (str, pt)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawString)[**drawString (str, x, y)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:drawString) | Desenha a string **str**para que a origem da linha de base apareça no ponto especificado. |
| [**getStringWidth (str)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getStringWidth) | Retorna a largura da string **str**quando exibida na fonte atual. |
| [**setFont (fonte)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:setFont) | Define uma nova fonte. |
| [**getFont ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getFont) | Retorna a fonte atual. |
| [**setColor (cor)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:setColor) | Define a cor usada para desenho. |
| [**getColor ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getColor) | Retorna a cor atual como uma string no formulário **"#rrggbb"**. |
| [**saveGraphicsState ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:saveGraphicsState) | Salva o estado do contexto gráfico. |
| [**restoreGraphicsState ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:restoreGraphicsState) | Restaura o estado gráfico da última chamada para **saveGraphicsState()**. |
| [**getWindowWidth ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getWindowWidth) | Retorna a largura da janela de gráficos em pixels. |
| [**getWindowHeight ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getWindowHeight) | Retorna a altura da janela de gráficos em pixels. |
| [**repintar ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:repaint) | Emite uma solicitação para atualizar a janela de gráficos. |
| [**pausa (milissegundos)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:pause) | Pausa para o número indicado de milissegundos. |
| [**waitForClick ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:waitForClick) | Espera um clique do mouse para ocorrer em qualquer lugar da janela. |
| [**setWindowTitle (título)**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:setWindowTitle) | Define o título da janela gráfica primária. |
| [**getWindowTitle ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:getWindowTitle) | Retorna o título da janela de gráficos primária. |
| [**exitGraphics ()**](https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html#Function:exitGraphics) | Fecha a janela gráfica e sai do aplicativo sem aguardar qualquer interação adicional do usuário. |

**Detalhe da função**

**vazio initGraphics ();**

**vazio initGraphics (int width, int height);**

Cria a janela de gráficos na tela. O primeiro formulário cria uma janela com um tamanho padrão de 500x300; O segundo permite que o cliente especifique o tamanho da janela. A chamada para **initGraphics**deve preceder qualquer saída do console ou chamadas para outras funções nesta interface.

Uso:

**initGraphics ();**

**initGraphics (largura, altura);**

**void drawArc (limites GRectangle, duplo começo, varredura dupla);**

**vazio drawArc (duplo x, duplo y, largura dupla, altura dupla,**

**duplo começo, varredura dupla);**

Desenha um arco elíptico inscrito em um retângulo. Os parâmetros **x**, **y**, **width**, e **height** (ou, equivalentemente, a **GRectangle** **bounds**) especifique as coordenadas e dimensões do rectângulo envolvente. O **start**parâmetro indica o ângulo em que o arco começa e é medido em graus no sentido anti-horário do eixo *x x* . Assim, um **start**ângulo de 0 indica um arco que começa ao longo da linha que corre a leste a partir do centro, um **start**ângulo de 135 começa ao longo da linha que corre a noroeste e um **start**ângulo de -90 começa ao longo da linha que corre para o sul. O **sweep**parâmetro indica a extensão do arco e também é medido em graus no sentido anti-horário. Um **sweep**ângulo de 90 define um quarto de círculo que se estende no sentido anti-horário do**start**ângulo e um **sweep**ângulo de -180 define um semicírculo que se estende no sentido horário.

Uso:

**drawArc (limites, início, varredura);**

**drawArc (x, y, largura, altura, início, varredura);**

**vazio fillArc (limites GRectangle, duplo começo, varredura dupla);**

**vazio fillArc (duplo x, duplo y, largura dupla, altura dupla,**

**duplo começo, varredura dupla);**

Preenche uma área em forma de cunha de um arco elíptico. Os parâmetros são interpretados da mesma maneira que aqueles para **drawArc**.

Uso:

**fillArc (limites, início, varredura);**

**fillArc (x, y, largura, altura, início, varredura);**

**vazio drawImage (string filename, GPoint pt);**

**vazio drawImage (string filename, double x, double y);**

**vazio drawImage (string filename, GRectangle bounds);**

**vazio drawImage (string filename, double x, double y,**

**largura dupla, altura dupla);**

Desenha a imagem do arquivo especificado com o canto superior esquerdo no ponto especificado. As formas da chamada que incluem os limites dimensionam a imagem para que ela se encaixe dentro do retângulo especificado.

Uso:

**drawImage (filename, pt);**

**drawImage (filename, x, y);**

**drawImage (filename, limites);**

**drawImage (nome do arquivo, x, y, largura, altura);**

**GRINDANGER getImageBounds (string filename);**

Retorna os limites da imagem contida no arquivo especificado. Apenas os componentes de largura e altura do retângulo são de interesse; Os componentes x e y são sempre 0.

Uso:

**GRectangle bounds = getImageBounds (filename);**

**void drawLine (GPoint p0, GPoint p1);**

**vazio drawLine (duplo x0, duplo y0, duplo x1, duplo y1);**

Desenha uma linha que liga os pontos especificados.

Uso:

**drawLine (p0, p1);**

**drawLine (x0, y0, x1, y1);**

**GPoint drawPolarLine (GPoint p0, double r, double theta);**

**GPoint drawPolarLine (duplo x0, duplo y0, double r, double theta);**

Desenha uma linha de comprimento **r**na direção **theta** do ponto inicial. O ângulo **theta**é medido em graus no sentido anti-horário do eixo *x* . A função retorna o ponto final da linha.

Uso:

**GPoint p1 = drawPolarLine (p0, r, theta);**

**GPoint p1 = drawPolarLine (x0, y0, r, theta);**

**void drawOval (limites GRectangle);**

**vazio drawOval (duplo x, duplo y, largura dupla, altura dupla);**

Desenha a moldura de um oval com os limites especificados.

Uso:

**drawOval (limites);**

**drawOval (x, y, largura, altura);**

**void fillOval (limites GRectangle);**

**vazio fillOval (duplo x, duplo y, largura dupla, altura dupla);**

Preenche o quadro de um oval com os limites especificados.

Uso:

**fillOval (limites);**

**fillOval (x, y, largura, altura);**

**vazio drawRect (limites GRectangle);**

**vazio drawRect (duplo x, duplo y, largura dupla, altura dupla);**

Desenha a moldura de um retângulo com os limites especificados.

Uso:

**drawRect (limites);**

**drawRect (x, y, largura, altura);**

**vazio fillRect (limites GRectangle);**

**preenchimento de preenchimento vazio (duplo x, duplo y, largura dupla, altura dupla);**

Preenche o quadro de um retângulo com os limites especificados.

Uso:

**fillRect (limites);**

**fillRect (x, y, largura, altura);**

**vazio drawPolygon (polígono do vetor <GPoint>);**

**vazio drawPolygon (vector <GPoint> polígono, GPoint pt);**

**void drawPolygon (polígono do vetor <GPoint>, duplo x, duplo y);**

Desenha o contorno do polígono especificado. Os parâmetros opcionais **pt** ou **x**e os **y**parâmetros mudam a origem do polígono para o ponto especificado.

Uso:

**desenharPolygon (polígono);**

**desenharPolygon (polígono, pt);**

**DesenhePolygon (polígono, x, y);**

**vazio preencherPolygon (vetor <polímero GPoint>);**

**vazio preencherPolygon (vector <GPoint> polígono, GPoint pt);**

**vazio preencherPolygon (polígono do vetor <GPoint>, duplo x, duplo y);**

Preenche o quadro do polígono especificado. Os parâmetros opcionais **pt** ou **x**e os **y**parâmetros mudam a origem do polígono para o ponto especificado.

Uso:

**preencher Polygon (polígono);**

**preencher Polygon (polígono, pt);**

**preencha Polygon (polígono, x, y);**

**vazio drawString (string str, GPoint pt);**

**vazio drawString (string str, double x, double y);**

Desenha a string **str**para que a origem da linha de base apareça no ponto especificado. O texto aparece na fonte e cor atual.

Uso:

**drawString (str, pt);**

**drawString (str, x, y);**

**double getStringWidth (string str);**

Retorna a largura da string **str**quando exibida na fonte atual.

Uso:

**largura dupla = getStringWidth (str);**

**vazio setFont (fonte de string);**

Define uma nova fonte. O **font**parâmetro é uma string na forma de *tamanho de estilo familiar* . Nessa cadeia, a *família* é o nome da família de fontes; *estilo* está ausente (indicando uma fonte simples) ou uma das cordas , , ou ; e o *tamanho* é um número inteiro que indica o tamanho do ponto. Se algum desses componentes for especificado como um asterisco, o valor existente será mantido. O parâmetro também pode ser uma seqüência de tais especificações separadas por ponto e vírgula, em que a primeira fonte disponível no sistema é usada. **--BoldItalicBoldItalicfont**

Uso:

**setFont (fonte);**

**string getFont ();**

Retorna a fonte atual.

Uso:

**string font = getFont ();**

**vazio setColor (cor da string);**

Define a cor usada para desenho. O **color**parâmetro é geralmente um dos nomes de cores pré-definidas de Java: **BLACK**, **BLUE**, **CYAN**, **DARK\_GRAY**, **GRAY**, **GREEN**, **LIGHT\_GRAY**, **MAGENTA**, **ORANGE**, **PINK**, **RED**,**WHITE**, ou **YELLOW**.

O caso das letras individuais no nome da cor é ignorado, assim como espaços e sublinhados, para que a cor Java **DARK\_GRAY**possa ser escrita como **"Dark Gray"**.

A cor também pode ser especificada como uma seqüência de caracteres na forma **"#rrggbb"**onde **rr**, **gg**e **bb**são pares de dígitos hexadecimais que indicam os componentes vermelho, verde e azul da cor.

Uso:

**setColor (cor);**

**string getColor ();**

Retorna a cor atual como uma string no formulário **"#rrggbb"**. Nessa cadeia, os valores **rr**, **gg**e **bb**são valores hexadecimais de dois dígitos que representam os componentes vermelho, verde e azul da cor, respectivamente.

Uso:

**string color = getColor ();**

**nulo saveGraphicsState ();**

Salva o estado do contexto gráfico. Esta função é usada em conjunto com **restoreGraphicsState()**para evitar a alteração do estado configurado pelo cliente.

Uso:

**saveGraphicsState ();**

**void restoreGraphicsState ();**

Restaura o estado gráfico da última chamada para **saveGraphicsState()**.

Uso:

**restoreGraphicsState ();**

**double getWindowWidth ();**

Retorna a largura da janela de gráficos em pixels.

Uso:

**largura dupla = getWindowWidth ();**

**double getWindowHeight ();**

Retorna a altura da janela de gráficos em pixels.

Uso:

**altura dupla = getWindowHeight ();**

**vazio repintar ();**

Emite uma solicitação para atualizar a janela de gráficos. Esta função é chamada automaticamente quando o programa faz uma pausa, aguarda um evento, aguarda a entrada do usuário no console ou termina. Como resultado, a maioria dos clientes nunca precisará chamar repintado explicitamente.

Uso:

**repintar ();**

**pausa vazia (milissegundos duplos);**

Pausa para o número indicado de milissegundos. Esta função é útil para a animação onde o movimento seria, de outra forma, muito rápido.

Uso:

**pausa (milissegundos);**

**vazio waitForClick ();**

Espera um clique do mouse para ocorrer em qualquer lugar da janela.

Uso:

**waitForClick ();**

**void setWindowTitle (título da string);**

Define o título da janela gráfica primária.

Uso:

**setWindowTitle (título);**

**string getWindowTitle ();**

Retorna o título da janela de gráficos primária.

Uso:

**string title = getWindowTitle ();**

**vazio exitGraphics ();**

Fecha a janela gráfica e sai do aplicativo sem aguardar qualquer interação adicional do usuário.

Uso:

**exitGraphics ();**