Highgui

Highgui é uma interface para "janelas" bem simples do OpenCV. Esta apostila mostra alguns exemplos de uso do Highgui.

Webcam: captura webcam e mostra na tela:

```
//webcam.cpp
#include <cekeikon.h>

int main(int argc, char** argv)
{ if (argc!=2 && argc!=4) {
    printf("CamTest: Testa webcam. Grava video com segundos.\n");
    printf("Webcam camId [nl nc]\n");
    printf("  Default: nl=480 nc=640\n");
    printf("  Ex: Webcam 0\n");
    erro("Erro: Numero de argumentos invalido");
  }

  int camid; convArg(camid,argv[1]);

  int nl=480;
  int nc=640;
  if (argc==4) {
    convArg(nl,argv[2]);
    convArg(nc,argv[3]);
  }

  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");
  vi.set(CV_CAP_PROP_FRAME_HEIGHT,nl);
  vi.set(CV_CAP_PROP_FRAME_WIDTH,nc);
  nl=vi.get(CV_CAP_PROP_FRAME_HEIGHT);
  nc=vi.get(CV_CAP_PROP_FRAME_WIDTH);
  printf("nl=%-4d nc=%-4d\n",nl,nc);

  Mat_<COR> a(nl,nc);
  namedWindow("janela");
  while (true)  {
    vi >> a;  //  get  a  new  frame  from  camera
    flip(a,a,1);
    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```

Observações:

- HighGUI só consegue atualizar janela durante a chamada de waitKey. Se nunca chamar waitKey, o seu programa não vai funcionar.
- Se o seu processamento demorar, convém colocar waitKey(1) em vários pontos intermediários do processamento. Caso contrário, o seu programa parecerá ter "travado".
- int ch=waitKey(0) espera indefinidamente até apertar uma tecla. O código da tecla apertada retorna a ch.
- int ch=waitKey(30) espera até apertar uma tecla ou passar 30 milissegundos. Se a tecla foi apertada, retorna o código da tecla. Caso contrário, retorna um número negativo.

Mouse: captura webcam e mostra eventos do mouse no console:

```
//mouse.cpp
#include <cekeikon.h>

void onMouse(int event, int x, int y, int flags, void* userdata)
{
  if  (event==EVENT_LBUTTONDOWN) {
    cout << "Apertou botao esquerda - posicao ("
         << x << ", " << y << ")" << endl;
  } else if (event==EVENT_RBUTTONDOWN) {
    cout << "Apertou botao direita  - posicao ("
         << x << ", " << y << ")" << endl;
  } else if ( event == EVENT_MOUSEMOVE ) {
    cout << "Mouse moveu na janela  - posicao ("
         << x << ", " << y << ")" << endl;
  }
}

int main(int argc, char** argv)
{ printf("Mouse: Captura webcam e mostra eventos do mouse.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  setMouseCallback("janela", onMouse);
  while (true)  {
    vi >> a;  //  get  a  new  frame  from  camera
    flip(a,a,1);
    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```

Sair: sai do programa com click do mouse:

```
//sair.cpp
#include <cekeikon.h>

void onMouse(int event, int x, int y, int flags, void* userdata)
{ bool* sair=(bool*)(userdata);
  if  (event==EVENT_LBUTTONDOWN) {
    if (0<=x && x<=79 && 0<=y && y<=79) *sair=true;
  }
}

int main(int argc, char** argv)
{ printf("Sair: Sai com click do mouse.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  // namedWindows("janela",0) -> permite redimensionar
  bool sair=false;
  setMouseCallback("janela", onMouse, (void*)(&sair));
  while (!sair)  {
    vi >> a;  //  get  a  new  frame  from  camera
    flip(a,a,1);
    retang(a,0,0,79,79,COR(0,0,255),2);
    putTxt(a,4,4,"SAIR",COR(0,0,255),2);
    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```

Usando variável global

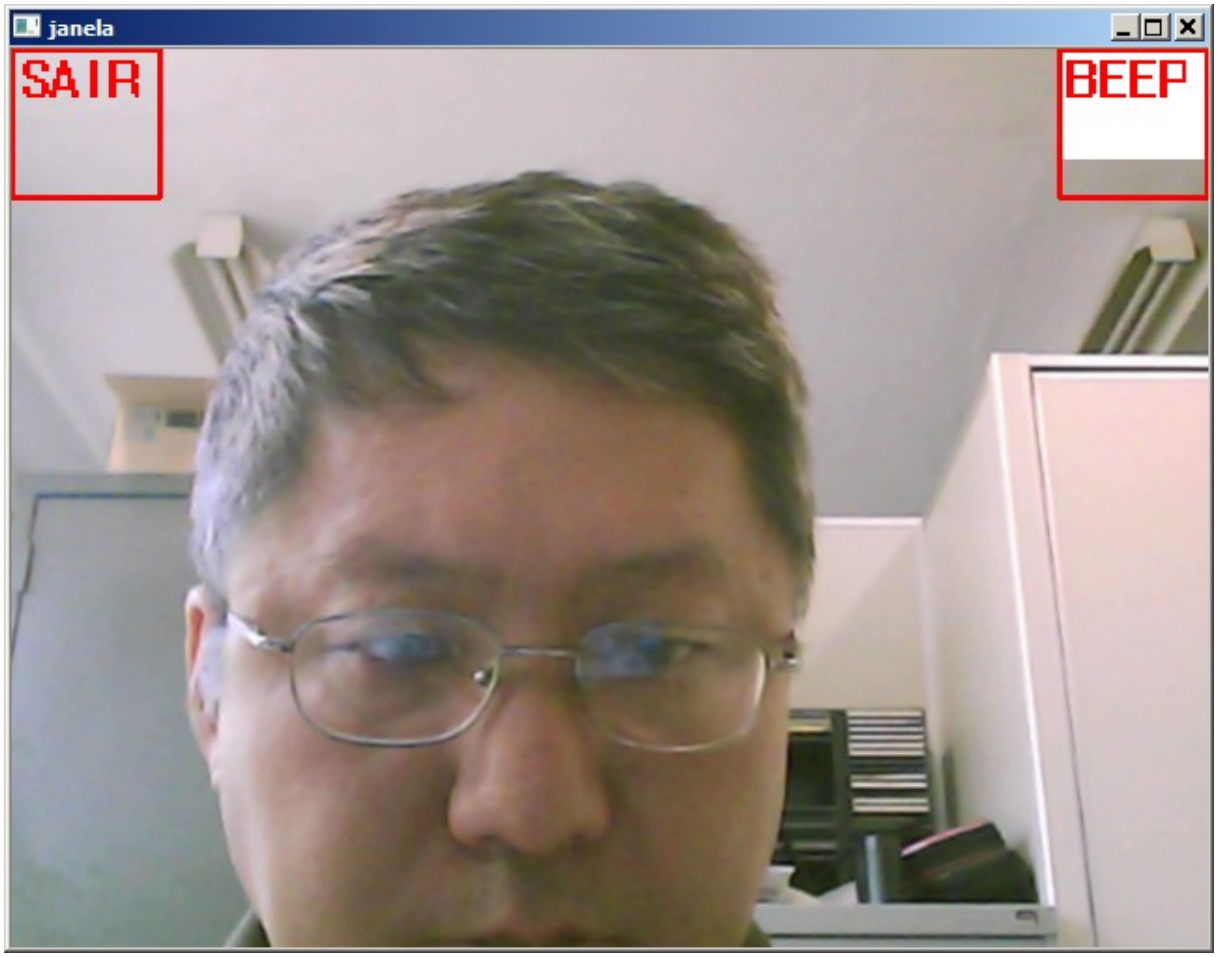
```
//sair2.cpp
#include <cekeikon.h>

bool sair;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if  (event==EVENT_LBUTTONDOWN) {
    if (0<=x && x<=79 && 0<=y && y<=79) sair=true;
  }
}

int main(int argc, char** argv)
{ printf("Sair: Sai com click do mouse.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  sair=false;
  setMouseCallback("janela", onMouse);
  while (!sair)  {
    vi >> a;  //  get  a  new  frame  from  camera
    flip(a,a,1);
    retang(a,0,0,79,79,COR(0,0,255),2);
    putTxt(a,4,4,"SAIR",COR(0,0,255),2);
    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```

Conta: sai do programa se o cursor ficar >=1.5s dentro do botão.

```
//conta.cpp
#include <cekeikon.h>

Point mouse(-1,-1);

void onMouse(int event, int x, int y, int flags, void* userdata)
{ mouse.x=x; mouse.y=y;
}

int main(int argc, char** argv)
{ printf("Conta: Sai do programa se cursor ficar 1.5s dentro do botao.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  bool sair=false;
  char estado='A';
  int t1;
  setMouseCallback("janela", onMouse);
  while (!sair)  {
    vi >> a;  //  get  a  new  frame  from  camera
    flip(a,a,1);

    if (estado=='A') {
      if (0<=mouse.y && mouse.y<80 && 0<=mouse.x && mouse.x<80) {
        estado='B'; t1=centseg();
      }
    } else if (estado=='B') {
      if (0<=mouse.y && mouse.y<80 && 0<=mouse.x && mouse.x<80) {
        int dt=centseg()-t1;
        int nl=cvRound(80.0*dt/150);
        fillRetang(a,0,0,nl-1,79,COR(255,255,255));
        if (dt>=150) sair=true;
      } else { estado='A'; }
    } else erro("Erro inesperado");
    retang(a,0,0,79,79,COR(0,0,255),2);
    putTxt(a,4,4,"SAIR",COR(0,0,255),2);

    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```

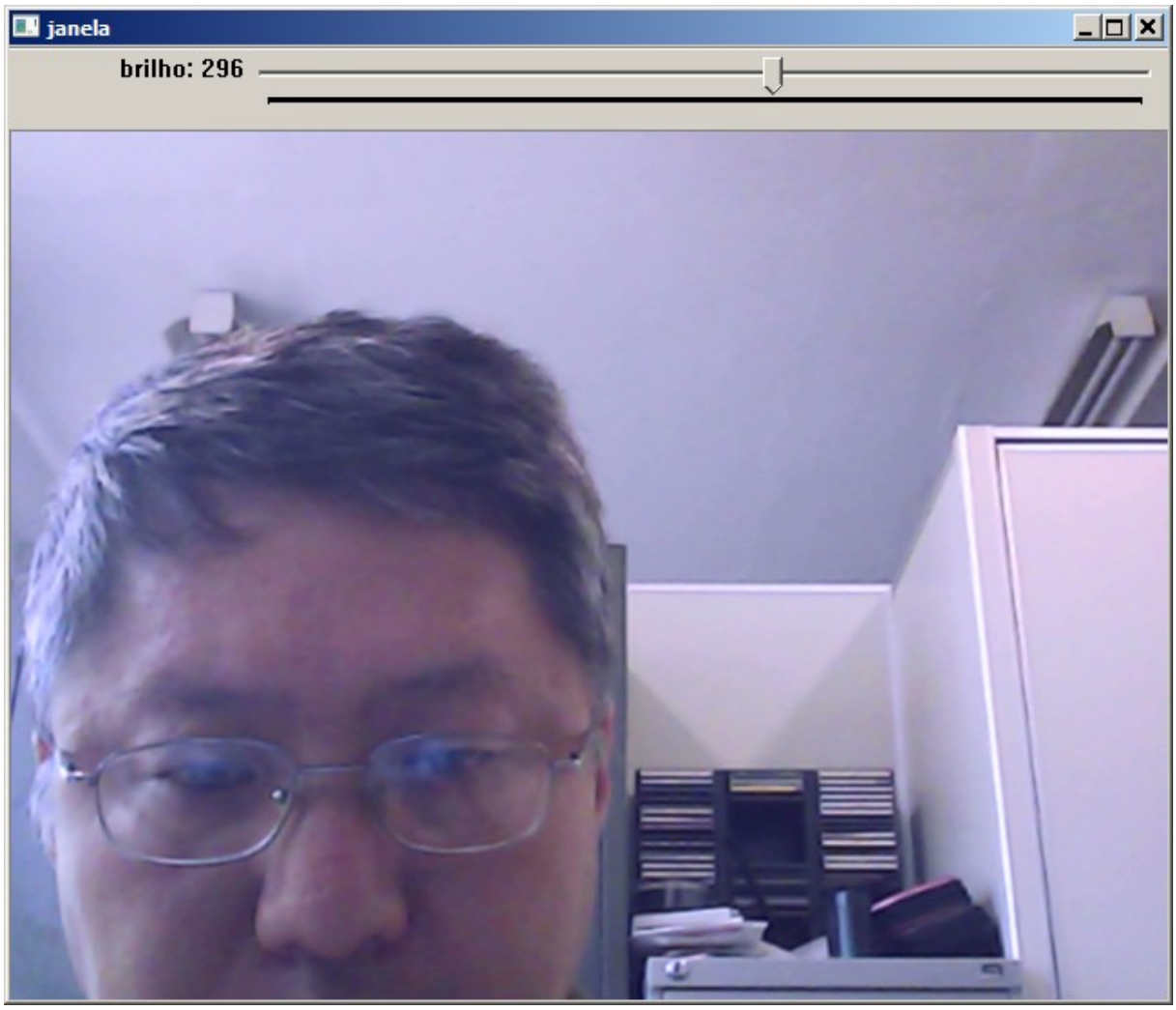
Conta2: Botões SAIR e BEEP. Utiliza programação orientada a objetos.

```
//conta2.cpp
#include <cekeikon.h>

Point mouse(-1,-1);

void onMouse(int event, int x, int y, int flags, void* userdata)
{ mouse.x=x; mouse.y=y;
}

class BOTAO {
 public:
  int espera;
  int li,ci,nl,nc;
  string rotulo;
  char estado;
  int t1;
  BOTAO(string protulo, int pli, int pci, int pnl, int pnc, int pespera)
    { rotulo=protulo; li=pli; ci=pci; nl=pnl; nc=pnc, espera=pespera; estado='A'; }
  bool apertou(Mat_<COR>& a);
};

bool BOTAO::apertou(Mat_<COR>& a)
{ bool b=false;
  if (estado=='A') {
    if (li<=mouse.y && mouse.y<li+nl && ci<=mouse.x && mouse.x<ci+nc) {
      estado='B'; t1=centseg();
    }
  } else if (estado=='B') {
    if (li<=mouse.y && mouse.y<li+nl && ci<=mouse.x && mouse.x<ci+nc) {
      int dt=centseg()-t1;
      int nlRetang=cvRound(nl*dt/150.0);
      fillRetang(a,li,ci,li+nlRetang-1,ci+nc-1,COR(255,255,255));
      if (dt>=espera) { b=true; estado='A'; }
    } else { estado='A'; }
  } else erro("Erro inesperado");
  retang(a,li,ci,li+nl-1,ci+nc-1,COR(0,0,255),2);
  putTxt(a,li+4,ci+4,rotulo,COR(0,0,255),2);
  return b;
}

int main(int argc, char** argv)
{ printf("Conta: Sai do programa se cursor ficar 1.5s dentro do botao.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  vi >> a;
  namedWindow("janela");
  BOTAO sair("SAIR",0,0,80,80,150);
  BOTAO beep("BEEP",0,a.cols-80,80,80,150);
  setMouseCallback("janela", onMouse);
  while (true)  {
    vi >> a;
    flip(a,a,1);
    if (sair.apertou(a)) break;
    if (beep.apertou(a)) MessageBeep(0);
    imshow("janela",a);
    if (waitKey(1)>=0) break;
  }
}
```


janela
SAIR
BEEP

Trackbar: Exemplo de uso de trackbar.

```
//trackbar.cpp
#include <cekeikon.h>

int main(int argc, char** argv)
{ printf("Trackbar: Permite somar valor na Mat_<COR> com trackbar.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  int brilho=256;
  createTrackbar("brilho", "janela", &brilho, 512);
  while (true)  {
    vi >> a;
    flip(a,a,1);
    a=a+(brilho-256);
    imshow("janela",a);
    if (waitKey(1)>=0)  break;
  }
}
```



Trackbar2: Permite controlar trackbar com mouse ou teclado.

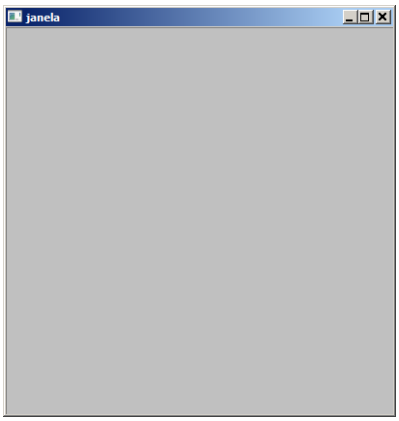
```
//trackbar2.cpp
#include <cekeikon.h>

int main(int argc, char** argv)
{ printf("Trackbar: Permite somar valor na Mat_<COR> com trackbar.\n");

  int camid=0;
  VideoCapture vi(camid);
  if (!vi.isOpened()) erro("Erro abertura webcam");

  Mat_<COR> a;
  namedWindow("janela");
  int brilho=256;
  createTrackbar("brilho", "janela", &brilho, 512);
  while (true)  {
    vi >> a;
    flip(a,a,1);
    a=a+(brilho-256);
    imshow("janela",a);
    int ch=waitKey(1);
    //if (ch>=0) printf("%d\n",ch);
    // botao direita
    if (ch==2555904) setTrackbarPos("brilho", "janela", brilho+1);
    // botao esquerda
    if (ch==2424832) setTrackbarPos("brilho", "janela", brilho-1);
    if (ch==27) break;
  }
}
```

Exemplo de uso do HighGUI para criar botões.

Os programas abaixo foram escritos originalmente para um outro curso. Deixo-os nesta apostila para mostrar a possibilidade de usar HighGUI para criar botões simples. Nesses programas, não uso Cekeikon, mas apenas OpenCV.

O nosso primeiro programa cria uma janela cinza e espera que o usuário aperte uma tecla para sair do programa.

```
// lampada01.cpp
// Cria janela cinza e espera apertar uma tecla para sair do programa
#include <opencv2/opencv.hpp>
using namespace cv;

int main()
{ printf("Sai do programa apertando uma tecla.\n");
  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  imshow("janela",a);
  waitKey(0);
}
```



A class Vec3b é um vetor com 3 bytes: blue, green e red. Os componentes de um Vec3b podem ser acessadas usando operador [].

A classe Mat_<Vec3b> é uma matriz que representa uma imagem colorida. Vamos usar variáveis deste tipo para armazenar o conteúdo da janela.

A declaração

```
Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
```

Cria uma imagem colorida com 400 linhas e 400 colunas e preenche-a com cor cinza clara.

A função imshow("janela",a); cria uma janela chamada *janela* e copia o conteúdo da imagem *a* para *janela*.

A função waitKey(0) espera infinitamente até que o usuário aperte alguma tecla. O programa não funciona sem esta função.

O nosso segundo programa sai do programa clicando o botão esquerdo do mouse. A janela cinza é igual à do programa anterior.

```
//Lampada02.cpp
//Sai do programa com um click esquerda do mouse:

#include <opencv2/opencv.hpp>
using namespace cv;
bool sair=false;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if  (event==EVENT_LBUTTONDOWN) {
    sair=true;
  }
}

int main()
{ printf("Sai do programa com click esquerda do mouse.\n");
  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  imshow("janela",a);
  while (!sair) {
    waitKey(30);
  }
}
```

A função waitKey(30) espera até que o usuário aperte alguma tecla ou que passe 30 milissegundos. É obrigatório chamar esta função periodicamente, pois é o único método em HighGUI que pode manipular eventos do sistema operacional. Sem esta função, os eventos do mouse não são capturados e a janela não é "refreshed".

Note que se colocar waitKey(0) em vez de waitKey(30) o programa não irá funcionar, pois esperará indefinidamente que o usuário aperte alguma tecla.

A função "callback" onMouse é chamada toda vez que houver alguma ação do mouse dentro da janela (tal como clicar ou mover mouse).

Repare que a variável "sair" está declarada como global. Há pessoas que defendem que não se deve usar variáveis globais (ou evitar ao máximo). Neste caso, é possível passar a variável "sair" como parâmetro através do "userdata".

Para ter uma ideia melhor de como a função "callback" onMouse é chamada, sugerimos que rode o seguinte programa:

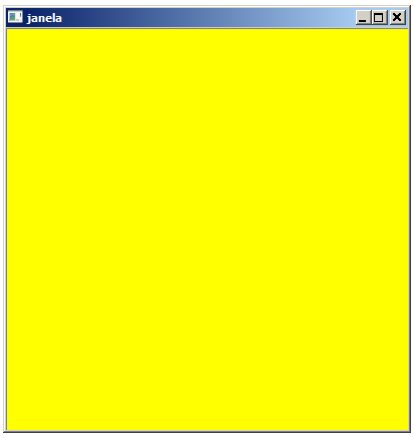
```
//mouse.cpp
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void onMouse(int event, int x, int y, int flags, void* userdata)
{
  if  (event==EVENT_LBUTTONDOWN) {
    cout << "Apertou botao esquerda - posicao ("
         << x << ", " << y << ")" << endl;
  } else if (event==EVENT_RBUTTONDOWN) {
    cout << "Apertou botao direita  - posicao ("
         << x << ", " << y << ")" << endl;
  } else if ( event == EVENT_MOUSEMOVE ) {
    cout << "Mouse moveu na janela  - posicao ("
         << x << ", " << y << ")" << endl;
  }
}

int main(int argc, char** argv)
{ printf("Mostra eventos do mouse.\n");

  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  setMouseCallback("janela", onMouse);
  imshow("janela",a);
  while (true)  {
    if (waitKey(1)>=0)  break;
  }
}
```

Exemplo de saída:

```
c:\diretorio>mouse
Mostra eventos do mouse.
Mouse moveu na janela  - posicao (5, 341)
Mouse moveu na janela  - posicao (10, 341)
Mouse moveu na janela  - posicao (16, 341)
Mouse moveu na janela  - posicao (18, 341)
Mouse moveu na janela  - posicao (20, 341)
Mouse moveu na janela  - posicao (21, 341)
Mouse moveu na janela  - posicao (22, 340)
Apertou botao esquerda - posicao (22, 340)
Mouse moveu na janela  - posicao (22, 340)
Apertou botao esquerda - posicao (22, 340)
Mouse moveu na janela  - posicao (22, 340)
Apertou botao direita  - posicao (22, 340)
Mouse moveu na janela  - posicao (22, 340)
Apertou botao direita  - posicao (22, 340)
Mouse moveu na janela  - posicao (22, 340)
```

O seguinte programa alterna entre janela preta e amarela com o click esquerdo do mouse.

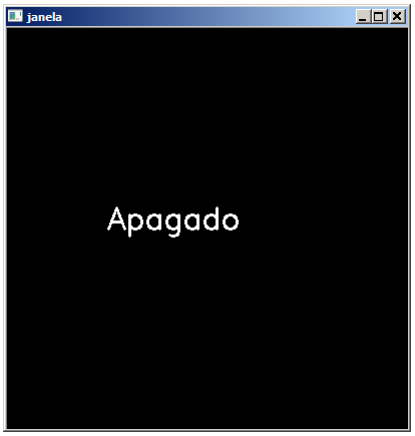
```
//Lampada03.cpp
//Sai do programa apertando uma tecla.
//Acende/apaga janela toda com click do mouse.

#include <opencv2/opencv.hpp>
using namespace cv;

bool aceso=false;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if  (event==EVENT_LBUTTONDOWN) {
    aceso=!aceso;
  }
}

int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Acende/apaga lampada com click do mouse.\n");

  Mat_<Vec3b> preto(400,400,Vec3b(0,0,0));
  Mat_<Vec3b> amarelo(400,400,Vec3b(0,255,255));
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  while (waitKey(30)<0) {
    if (aceso) imshow("janela",amarelo);
    else       imshow("janela",preto);
  }
}
```

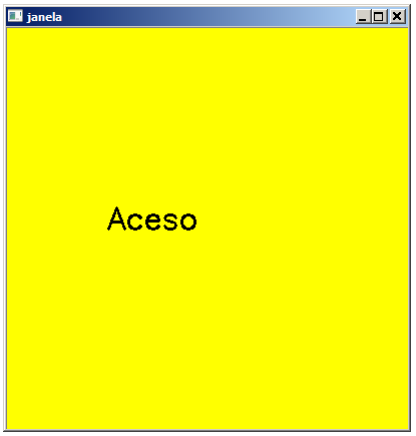


Nota: O programa acima é ineficiente, pois todo o conteúdo da imagem “preto” ou “amarelo” é copiado a cada 30ms para a janela, mesmo que o usuário não tenha clicado mouse.

Vamos resolver o “problema da ineficiência” do programa anterior e também vamos escrever alguma coisa na janela.

```
//Lampada04.cpp
//Acende/apaga janela toda clicando. Escreve o estado.

#include <opencv2/opencv.hpp>
using namespace cv;

bool aceso=false;
bool clicou=false;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if  (event==EVENT_LBUTTONDOWN) {
    aceso=!aceso;
    clicou=true;
  }
}

int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Acende/apaga lampada com click.\n");

  Mat_<Vec3b> preto(400,400,Vec3b(0,0,0));
  putText(preto, "Apagado", Point(100,200), FONT_HERSHEY_SIMPLEX, 1, Scalar(255,255,255), 2);
  Mat_<Vec3b> amarelo(400,400,Vec3b(0,255,255));
  putText(amarelo, "Aceso", Point(100,200), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  imshow("janela",preto);
  while (waitKey(30)<0) {
    if (clicou) {
      if (aceso) imshow("janela",amarelo);
      else       imshow("janela",preto);
      clicou=false;
    }
  }
}
```



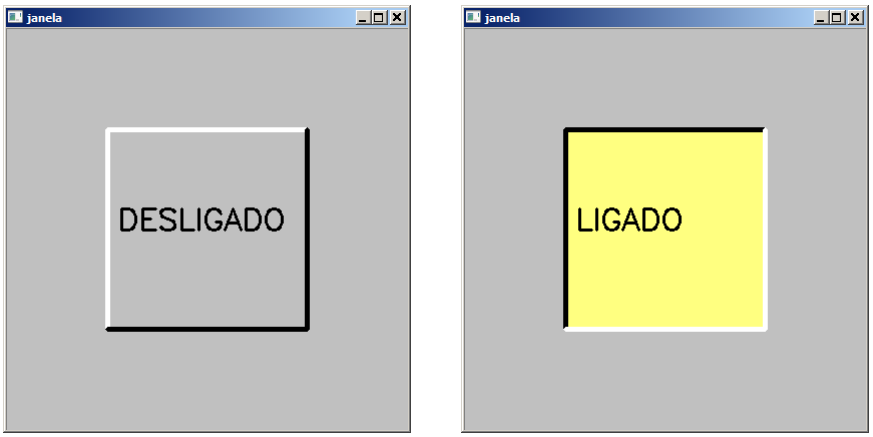


Note que o conteúdo da imagem só é copiado para janela quando o usuário clicar mouse.

A posição (x,y)=(0,0) da imagem fica no canto superior esquerdo.

O programa abaixo cria um botão liga-desliga.

```
//Lampada05.cpp
//Cria um botao liga-desliga.

#include <opencv2/opencv.hpp>
using namespace cv;

bool desligado=true;
bool clicou=true;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if (event==EVENT_LBUTTONDOWN && 100<=x && x<=299 && 100<=y && y<=299) {
    desligado=!desligado;
    clicou=true;
  }
}

int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Muda o estado do botao com um click.\n");

  Mat_<Vec3b> cinza(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  imshow("janela",cinza);
  while (waitKey(30)<0) {
    if (clicou) {
      if (desligado) {
        rectangle(cinza, Point(100,100), Point(299,299), Scalar(192,192,192), CV_FILLED);
        line(cinza, Point(100,100), Point(100,299), Scalar(255,255,255), 4);
        line(cinza, Point(100,100), Point(299,100), Scalar(255,255,255), 4);
        line(cinza, Point(299,299), Point(100,299), Scalar(0,0,0), 4);
        line(cinza, Point(299,299), Point(299,100), Scalar(0,0,0), 4);
        putText(cinza, "DESLIGADO", Point(110,200),FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
      } else {
        rectangle(cinza, Point(100,100), Point(299,299), Scalar(128,255,255), CV_FILLED);
        line(cinza, Point(100,100), Point(100,299), Scalar(0,0,0), 4);
        line(cinza, Point(100,100), Point(299,100), Scalar(0,0,0), 4);
        line(cinza, Point(299,299), Point(100,299), Scalar(255,255,255), 4);
        line(cinza, Point(299,299), Point(299,100), Scalar(255,255,255), 4);
        putText(cinza, "LIGADO", Point(110,200),FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
      }
      imshow("janela",cinza);
      clicou=false;
    }
  }
}
```



A variável clicou deve ser true no início do programa. Caso contrário, o botão não será desenhado no início do programa.

Para poder criar vários botões facilmente, vamos criar a classe Botao. As janelas são iguais às do programa anterior.

```
//Lampada06.cpp
//Cria a classe Botao.

#include <opencv2/opencv.hpp>
using namespace cv;

bool clicou=true;
int xm,ym;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if (event==EVENT_LBUTTONDOWN) {
    clicou=true; xm=x; ym=y;
  }
}

class Botao {
  public:
    Point p,q;
    string st0,st1;
    bool ligado,primeiraVez;
    void cria(Point _p, Point _q, string _st0, string _st1, bool _ligado=false);
    void testaDesenha(Mat_<Vec3b>& a);
};

void Botao::cria(Point _p, Point _q, string _st0, string _st1, bool _ligado)
{ p=_p; q=_q; st0=_st0; st1=_st1; ligado=!_ligado; primeiraVez=true; }

void Botao::testaDesenha(Mat_<Vec3b>& a)
{ if (primeiraVez || (p.x<=xm && xm<=q.x && p.y<=ym && ym<=q.y)) {
    if (ligado) {
      ligado=false;
      rectangle(a, p,q, Scalar(192,192,192), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, p, Point(q.x,p.y), Scalar(255,255,255), 4);
      line(a, q, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, q, Point(q.x,p.y), Scalar(0,0,0), 4);
      putText(a, st0, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    } else {
      ligado=true;
      rectangle(a, p,q, Scalar(128,255,255), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, p, Point(q.x,p.y), Scalar(0,0,0), 4);
      line(a, q, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, q, Point(q.x,p.y), Scalar(255,255,255), 4);
      putText(a, st1, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    }
    primeiraVez=false;
  }
}

int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Liga e desliga o botao com click.\n");

  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  imshow("janela",a);
  Botao b;
  b.cria(Point(100,100),Point(299,299),"DESLIGADO","LIGADO");
  while (waitKey(30)<0) {
    if (clicou) {
      b.testaDesenha(a);
      imshow("janela",a);
      clicou=false;
    }
  }
}
```

Nota: Todas as variáveis e funções da classe Botao foram declaradas como públicas. Algumas pessoas acham que isto é uma forma ruim de programar.

Vamos usar a classe Botao para criar vários botões dentro da janela.

```
//Lampada07
//Utiliza a classe Botao para criar varios botoes.

#include <opencv2/opencv.hpp>
using namespace cv;

bool clicou=true;
int xm,ym;

void onMouse(int event, int x, int y, int flags, void* userdata)
{ if (event==EVENT_LBUTTONDOWN) {
    clicou=true; xm=x; ym=y;
  }
}

class Botao {
  public:
    Point p,q;
    string st0,st1;
    bool ligado,primeiraVez;
    void cria(Point _p, Point _q, string _st0, string _st1="", bool _ligado=false);
    void testaDesenha(Mat_<Vec3b>& a);
};

void Botao::cria(Point _p, Point _q, string _st0, string _st1, bool _ligado)
{ p=_p; q=_q; st0=_st0;
  if (_st1!="") st1=_st1; else st1=_st0;
  ligado=!_ligado; primeiraVez=true;
}

void Botao::testaDesenha(Mat_<Vec3b>& a)
{ if (primeiraVez || (p.x<=xm && xm<=q.x && p.y<=ym && ym<=q.y)) {
    if (ligado) {
      ligado=false;
      rectangle(a, p,q, Scalar(192,192,192), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, p, Point(q.x,p.y), Scalar(255,255,255), 4);
      line(a, q, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, q, Point(q.x,p.y), Scalar(0,0,0), 4);
      putText(a, st0, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    } else {
      ligado=true;
      rectangle(a, p,q, Scalar(128,255,255), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, p, Point(q.x,p.y), Scalar(0,0,0), 4);
      line(a, q, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, q, Point(q.x,p.y), Scalar(255,255,255), 4);
      putText(a, st1, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    }
    primeiraVez=false;
  }
}
```

```
int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Liga e desliga os botoes com click.\n");

  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  setMouseCallback("janela", onMouse, 0);
  imshow("janela",a);
  Botao b1; b1.cria(Point( 10, 10),Point(189,189),"BOTAO1");
  Botao b2; b2.cria(Point(210, 10),Point(389,189),"BOTAO2-D","BOTAO2-L");
  Botao b3; b3.cria(Point( 10,210),Point(189,389),"BOTAO3");
  Botao b4; b4.cria(Point(210,210),Point(389,389),"BOTAO4-D","BOTAO4-L");
  while (waitKey(30)<0) {
    if (clicou) {
      b1.testaDesenha(a);
      b2.testaDesenha(a);
      b3.testaDesenha(a);
      b4.testaDesenha(a);
      imshow("janela",a);
      clicou=false;
    }
  }
}
```

Para organizar mais o programa vamos criar também a classe Mouse. Porém, não é possível colocar a função onMouse dentro da classe Mouse, pois onMouse é uma função especial. As janelas são iguais às do programa anterior.

```
//Lampada08.cpp
//Cria classe Mouse.

#include <opencv2/opencv.hpp>
using namespace cv;

struct Mouse {
  bool clicou=true;
  int x,y;
};

void onMouse(int event, int x, int y, int flags, void* userdata)
{ Mouse* mouse=(Mouse*)userdata;
  if (event==EVENT_LBUTTONDOWN) {
    (*mouse).clicou=true; (*mouse).x=x; (*mouse).y=y;
  }
}

class Botao {
  public:
    Point p,q;
    string st0,st1;
    bool ligado,primeiraVez;
    Botao(Point _p, Point _q, string _st0, string _st1="", bool _ligado=false);
    void testaDesenha(Mat_<Vec3b>& a, Mouse mouse);
};

Botao::Botao(Point _p, Point _q, string _st0, string _st1, bool _ligado)
{ p=_p; q=_q; st0=_st0;
  if (_st1!="") st1=_st1; else st1=_st0;
  ligado=!_ligado; primeiraVez=true;
}

void Botao::testaDesenha(Mat_<Vec3b>& a, Mouse mouse)
{ if (primeiraVez || (p.x<=mouse.x && mouse.x<=q.x && p.y<=mouse.y && mouse.y<=q.y)) {
    if (ligado) {
      ligado=false;
      rectangle(a, p,q, Scalar(192,192,192), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, p, Point(q.x,p.y), Scalar(255,255,255), 4);
      line(a, q, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, q, Point(q.x,p.y), Scalar(0,0,0), 4);
      putText(a, st0, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    } else {
      ligado=true;
      rectangle(a, p,q, Scalar(128,255,255), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, p, Point(q.x,p.y), Scalar(0,0,0), 4);
      line(a, q, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, q, Point(q.x,p.y), Scalar(255,255,255), 4);
      putText(a, st1, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    }
    primeiraVez=false;
  }
}
```

```
int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Liga e desliga os botoes com click.\n");

  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  Mouse mouse;
  setMouseCallback("janela", onMouse, &mouse);
  imshow("janela",a);
  Botao b1(Point( 10, 10),Point(189,189),"BOTAO1");
  Botao b2(Point(210, 10),Point(389,189),"BOTAO2-D","BOTAO2-L");
  Botao b3(Point( 10,210),Point(189,389),"BOTAO3");
  Botao b4(Point(210,210),Point(389,389),"BOTAO4-0","BOTAO4-1");
  while (waitKey(30)<0) {
    if (mouse.clicou) {
      b1.testaDesenha(a,mouse);
      b2.testaDesenha(a,mouse);
      b3.testaDesenha(a,mouse);
      b4.testaDesenha(a,mouse);
      imshow("janela",a);
      mouse.clicou=false;
    }
  }
}
```

Nota 1: “struct” é o mesmo que “class public”.

Nota 2: Para evitar de usar variável global (o que alguns autores condenam), a variável mouse é passado como parâmetro.

Vamos criar a class Lampada. Com as classes Botao e Lampada, é possível simular liga-desliga das lâmpadas.

```
//Lampada09.cpp
//Cria classe Lampada.

#include <opencv2/opencv.hpp>
using namespace cv;

struct Mouse {
  bool clicou=true;
  int x,y;
};

void onMouse(int event, int x, int y, int flags, void* userdata)
{ Mouse* mouse=(Mouse*)userdata;
  if (event==EVENT_LBUTTONDOWN) {
    (*mouse).clicou=true; (*mouse).x=x; (*mouse).y=y;
  }
}

class Botao {
  public:
    Point p,q;
    string st0,st1;
    bool primeiraVez;
    bool ligado;
    Botao(Point _p, Point _q, string _st0, string _st1="", bool _ligado=false);
    void testaDesenha(Mat_<Vec3b>& a, Mouse mouse);
};

Botao::Botao(Point _p, Point _q, string _st0, string _st1, bool _ligado)
{ p=_p; q=_q; st0=_st0;
  if (_st1!="") st1=_st1; else st1=_st0;
  ligado=!_ligado; primeiraVez=true;
}

void Botao::testaDesenha(Mat_<Vec3b>& a, Mouse mouse)
{ if (primeiraVez || (p.x<=mouse.x && mouse.x<=q.x && p.y<=mouse.y && mouse.y<=q.y)) {
    if (ligado) {
      ligado=false;
      rectangle(a, p,q, Scalar(160,160,160), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, p, Point(q.x,p.y), Scalar(255,255,255), 4);
      line(a, q, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, q, Point(q.x,p.y), Scalar(0,0,0), 4);
      putText(a, st0, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    } else {
      ligado=true;
      rectangle(a, p,q, Scalar(160,160,160), CV_FILLED);
      line(a, p, Point(p.x,q.y), Scalar(0,0,0), 4);
      line(a, p, Point(q.x,p.y), Scalar(0,0,0), 4);
      line(a, q, Point(p.x,q.y), Scalar(255,255,255), 4);
      line(a, q, Point(q.x,p.y), Scalar(255,255,255), 4);
      putText(a, st1, Point(p.x+5,(p.y+q.y)/2), FONT_HERSHEY_SIMPLEX, 1, Scalar(0,0,0), 2);
    }
    primeiraVez=false;
  }
}

class Lampada {
    Point p;
    int r;
  public:
    bool acesa;
    Lampada(Point _p, int _r) { p=_p; r=_r; }
    void desenha(Mat_<Vec3b>& a);
};

void Lampada::desenha(Mat_<Vec3b>& a)
{ if (acesa) circle(a,p,r,Scalar(0,255,255),-1);
  else       circle(a,p,r,Scalar(0,0,0),-1);
}
```

```
int main()
{ printf("Sai do programa apertando uma tecla.\n");
  printf("Muda estado da lampada com click no botao.\n");

  Mat_<Vec3b> a(400,400,Vec3b(192,192,192));
  namedWindow("janela");
  Mouse mouse;
  setMouseCallback("janela", onMouse, &mouse);
  imshow("janela",a);
  Botao botao1(Point( 10, 10),Point(189,189),"BOTAO-0","BOTAO-1");
  Lampada lampada1(Point(300,100),80);
  Botao botao2(Point( 10,210),Point(189,389),"BOTAO-0","BOTAO-1");
  Lampada lampada2(Point(300,300),80);
  while (waitKey(30)<0) {
    if (mouse.clicou) {
      botao1.testaDesenha(a,mouse);
      lampada1.acesa=botao1.ligado;
      lampada1.desenha(a);
      botao2.testaDesenha(a,mouse);
      lampada2.acesa=botao2.ligado;
      lampada2.desenha(a);
      imshow("janela",a);
      mouse.clicou=false;
    }
  }
}
```