Vitor Amadeu Souza

Antologia

de projetos com TFT programado no Arduino em

Artes Visuais

Volume IV

© 2020 by Cerne Tecnologia e Treinamento Ltda.

© 2020 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Janeiro de 2020

Direitos reservados por:
Cerne Tecnologia e Treinamento Ltda
Produção: Cerne Tecnologia e Treinamento
E-mail da Empresa: cerne@cerne-tec.com.br
Home Page: www.cerne-tec.com.br.com.br
Atendimento ao Consumidor: sac@cerne-tec.com.br
Contato com o Autor: vitor@cerne-tec.com.br



FEITO NO BRASIL

Dedicatória

Como nos meus outros livros, dedico este livro a minha querida esposa Renata Leal.

"Para quem tudo perde, ainda resta Deus."

Alfred de Musset

Cerne Tecnologia

A Cerne Tecnologia tem uma equipe preparada para desenvolvimento de projetos eletrônicos em diversas áreas: Médica, Entretenimento, Industrial, Robótica, Científica, Automobilística, Aeronáutica, etc. Trabalhamos com tecnologia microcontrolada usando o PIC, ARM, AVR, 8051, dsPIC, PIC24, PIC32 além do Arduino, Raspberry, Beaglebone etc. Desenvolvemos o projeto desde sua concepção até a entrega do produto final, passando pelas etapas de esquema elétrico, protótipo e desenvolvimento de circuito impresso.

Desenvolvemos aplicativos para smartphones/tablets Android, iOS, Blackberry, Windows Phone e no desenvolvimento de softwares a nível PC para plataforma Windows, usando ferramentas como o Visual Basic, C# e C++.

Atuamos na parte de montagem de placas, onde podemos fornecer ambos os serviços de desenvolvimento de projetos e produção ou apenas um destes.

Desenvolvemos esquemas elétricos e layout de PCI, tanto em tecnologia convencional como SMD.

Temos a flexibilidade de customizar um de nossos produtos, de modo a atender a uma necessidade específica do cliente, tornando o custo de desenvolvimento menor se comparado a construção de um projeto desde a sua fase inicial.

Desenvolvemos e fornecemos kits didáticos para diversos microcontroladores além de apostilas, livros e e-books.

Na hora de desenvolver um projeto ou equipar seu laboratório não hesite em nos contatar. Entre em contato conosco através do endereço cerne-tec.com.br para obter mais informações.

Cerne
Conhecimento para o Desenvolvimento

cerne-tec.com.br

Sumário

01 – Metodologia de desenvolvimento 9

02 – Tela TFT 10

03 – Marseus van Schrieck 24

04 – Otto Stark 28

05 – Otto van Veen 32

06 – Pál Szinyei Merse 36

07 – Paleografia 40

08 – Paolo Uccello 42

09 – Paris Bordone 46

10 – Paul Cézanne 50

11 – Paul Delaroche 54

12 – Eugène-Henri-Paul Gauguin 58

13 – Pedro Américo 62

14 – Pedro Campaña 66

15 – Península 70

16 – Philip Wilson Steer 73

17 – Philipp Otto Runge 77

18 – Pictograma 81

19 – Piero del Pollaiolo 85

20 – Piero Della Francesca 89

21 – Piero di Cosimo 93

22 – Pierre-Auguste Renoir 97

23 – Pieter Bruegel 101

24 – Pieter Coecke van Aelst 105

25 – Pietro de Cortona 109

26 – Pietro Perugino 113

27 – Pinacoteca 117

28 – Pensamentos de Pitágoras 120

29 – Pensamentos de Platão 123

30 – Polidoro de Caravaggio....126

31 – Pomponio Amalteo....130

32 – Ralph Albert Blakelock....134

33 – Rembrandt....138

34 – René Descartes....142

35 – Richard Dadd....145

36 – Robert William Vonnoh....149

37 – Rodolfo Amoedo....153

38 – Rogier van der Weyden....157

39 – Romênia....161

40 – Rondônia....163

41 – Roraima....165

42 – Rosa Bonheur....167

43 – Pensamentos de Rousseau....171

44 – Ruanda....174

45 – Rússia....176

46 – Samoa....178

47 – Samuel Palmer....180

48 – San Marino....184

49 – Sandro Botticelli....186

50 – Santa Catarina....189

51 – Santi di Tito....191

52 – Pensamentos de São Francisco de Assis....195

53 – Vexilologia de São Paulo....198

54 – Sebastián López de Arteaga....200

55 – Sebastian Vrancx....204

56 – Sebastiano del Piombo....207

57 – Senegal....211

58 – Vexilologia de Sergipe....213

59 – Sigilografia....215

60 – Simbologia....219

61 – Pensamentos de Sócrates **223**

62 – Sofonisba Anguissola **226**

63 – Stefano di Giovanni **230**

64 – Taddeo Gaddi **234**

65 – Théodore Chassériau **238**

66 – Jean-Louis André Théodore Géricault **242**

67 – Thomas Cole **246**

68 – Thomas Cowperthwait Eakins **250**

69 – Thomas Gainsborough **254**

70 – Ticiano **258**

71 – Timoteo Viti **262**

72 – Tintoretto **266**

73 – Tomás José da Anunciação **270**

74 – Umberto Boccioni **274**

75 – Valentin Alexandrovich Serov **278**

76 – Vicente Juan Masip **282**

77 – Victor Meirelles **286**

78 – Van Gogh **290**

79 – Vitral **294**

80 – Vittore Carpaccio **297**

81 – Pensamentos de Voltaire **301**

82 – Washington Allston **304**

83 – William Blake **308**

84 – William Dobson **312**

85 – William Etty **316**

86 – William Hogarth **320**

87 – Pensamentos de William Shakespeare **324**

88 – Joseph Mallord William Turner **327**

89 – William-Adolphe Bouguereau **331**

90 – Winslow Homer **335**

91 – Władysław Podkowiński **339**

92 – Xilogravura.....343

Capítulo 1
Metodologia de desenvolvimento

1. Introdução

Esta literatura é uma continuação da obra *Arduino – Prático e Objetivo* (2011) e *Programação para Arduino – Avançado* (2014) do mesmo autor e editora, onde o objetivo é controlar uma tela de TFT usando o Arduino para em seguida apresentar imagens relacionadas listadas no sumário através de uma tela TFT.

A placa didática utilizada foi a Arduino UNO em conjunto com um display TFT que faz uso do controlador ST7735, onde tais kits e acessórios estão à venda no site www.cerne-tec.com.br.

Capítulo 2
Tela TFT

1. TFT

O TFT (Thin-Film-Transistor liquid-crystal display) utilizado nesta experiência é o modelo cuja imagem está apresentada a seguir.



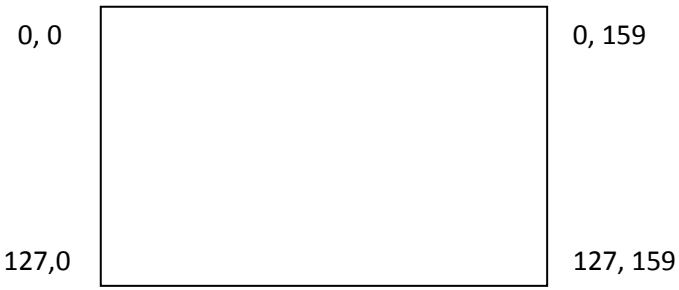
Fonte: https://www.arduino.cc/en/Guide/TFT#toc3

Este display possui uma tela de 1,8º e possui resolução 160x128 pixels (HxV). Utiliza um controlador, cujo código é ST7735 e a alimentação é feita em 5V.

Outros modelos podem ser utilizados, desde que o controlador seja compatível. O software Arduino a partir da versão 1.0.5 já dispõe de bibliotecas para acessá-lo.

A tela de TFT citada possui encaixe para cartão SD CARD, onde imagens podem ser gravadas no formato bitmap (bmp) para serem carregadas na tela. Para isso, faz-se necessário habilitar a conexão através do barramento SPI com o cartão. Este display pode ser encaixado como um shield a placa Arduino.

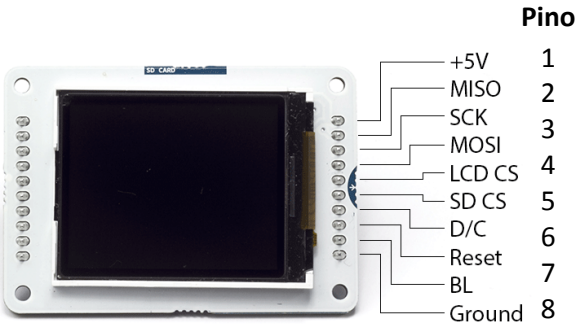
Por ter uma resolução de 160x128 pixels, podemos ver esta tela como um plano cartesiano, cujos pontos estão apresentados a seguir.



A resolução de cada pixel do display é de 16 bits, sendo 5 bits para R (Red-Vermelho) e B (Blue-Azul) e 8 para G (Green-Verde).

2. Pinagem

A pinagem deste display está apresentada abaixo.



Fonte: https://www.arduino.cc/en/Guide/TFT#toc3

A tela é alimentada em 5V através dos pinos 1 e 8. Os pinos MISO (2), SCK(3), MOSI(4) fazem a comunicação SPI entre o Arduino e o display e cartão SD. A seleção de comunicação se será feita com o TFT ou SD depende dos pinos digitais LCD CS (5) e SD CS (6). Os pinos DC e RESET também devem ser utilizados para conectar o TFT ao Arduino.

Quaisquer pinos podem ser utilizados para CS, DC e RESET, no entanto, para a comunicação SPI recomenda-se utilizar os pinos default para esta comunicação, disponível no conector ICSP da placa Arduino em razão da sua maior velocidade de transação.

3. Alterando a cor de fundo

Este experimento troca a cor da tela (background) de acordo com o parâmetro passado através do método `background` após a

inicialização do objeto `tela`. Acompanhe a seguir a listagem do código.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,0,0);
}

void loop()
{

}
```

Inicialmente são feitas as inclusões referentes às bibliotecas de comunicação SPI e TFT. Em seguida, os pinos referentes à conexão com a tela gráfica como LCD_RS, DC e RESET são definidos no programa e o objeto `tela` é criado em função dos pinos definidos anteriormente.

No bloco `setup()` o TFT é inicializado pelo método begin e a cor de fundo do display é alterado pelo método `background()` já que o parâmetro passado é 0,0,0. Tal parâmetro refere-se às cores

R, G e B (vermelho, verde e azul) onde cada parâmetro pode ser alterado de 0 a 255 em que quanto mais próximo de 255 maior a intensidade da cor especificada e quanto mais próximo de 0 maior é a sua ausência. Ao passar o parâmetro 0,0,0 foi informada a ausência das três cores fundamentais, logo a tela fica escura.

Para deixar a tela com o fundo branco, teste o código a seguir.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
}

void loop()
{

}
```

Diferente do exemplo anterior este deixa todas as cores presentes, onde o resultado é o fundo branco. Acompanhe a seguir o programa para deixar toda a tela vermelha.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,0,0);
}

void loop()
{

}
```

O próximo exemplo deixa a tela verde.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,255,0);
}

void loop()
{
```

```
}
```

E o próximo exemplo com a cor azul.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,0,255);
}

void loop()
{

}
```

4. Desenhando linhas

Ao criar o objeto tela uma série de métodos são disponibilizados. Uma delas é a `line(`x_i`,`y_i`,`x_f`,`y_f`)` que possibilita o desenho de uma linha de acordo o ponto inicial `(`x_i`,`y_i`)` e final `(`x_f`,`y_f`)` informado como parâmetro. O método `stroke(R,G,B)` permite a alteração da cor no qual a linha será impressa.

Acompanhe a seguir o programa para desenhar uma linha diagonal vermelha na tela do TFT.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(255,0,0);
  tela.line(0,0,159,127);
}

void loop()
{

}
```

O próximo exemplo desenha uma linha horizontal verde na tela do TFT.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,255,0);
  tela.line(0,127,159,127);
}

void loop()
{

}
```

5. Desenhando retângulos

O método `rect(`x_i`,`y_i`,largura,altura)` possibilita o desenho de um retângulo a partir do ponto inicial `(`x_i`,`y_i`)` de acordo com a `largura` e `altura` especificadas como parâmetro. O método `stroke(R,G,B)` permite a alteração da cor da linha que envolve o retângulo enquanto o método `fill(R,G,B)` a sua cor interna.

Acompanhe a seguir o programa para desenhar um retângulo com cor interna vermelha e linha de contorno azul.

```
#include <SPI.h>
#include <TFT.h>
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,0,255);
  tela.fill(255,0,0);
  tela.rect(0,0,50,50);
}

void loop()
{

}
```

6. Desenhando círculos

O método `circle(`x_i`,`y_i`,raio)` possibilita o desenho de um círculo a partir do ponto inicial (x_i, y_i) de acordo com o `raio` especificado no parâmetro. O método `stroke(R,G,B)` permite a alteração da cor da linha que envolve o círculo enquanto o método `fill(R,G,B)` a sua cor interna.

Acompanhe a seguir o programa para desenhar um círculo com cor interna vermelha e linha de contorno azul.

```
#include <SPI.h>
#include <TFT.h>
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,0,255);
  tela.fill(255,0,0);
  tela.circle(50,50,20);
}

void loop()
{

}
```

7. Imprimindo textos

O método `text(texto,`x_i`,`y_i`)` possibilita escrever um `texto` a partir do ponto inicial `(`x_i`,`y_i`)`. O tamanho da letra por default é de 5 pixels de comprimento e 8 de altura.

Acompanhe a seguir o programa para escrever um texto na tela TFT.

```
#include <SPI.h>
#include <TFT.h>


#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
```

```
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.text("Cerne Tec",0,50);
}

void loop()
{

}
```

O método `setTextSize(tamanho)` possibilita aumentar a altura da letra de 10 em 10 pixels. Por exemplo, caso `setTextSize(1), setTextSize(2) e setTextSize(3)` a altura será de 10, 20 e 30 pixels respectivamente, como mostra o próximo exemplo.

```
#include <SPI.h>
#include <TFT.h>

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(255,0,0);
  tela.setTextSize(1);
  tela.text("Cerne",0,0);
  tela.setTextSize(2);
```

```
  tela.text("Tecnologia",0,20);
  tela.setTextSize(1);
  tela.text("cerne-tec.com.br",0,50);
}

void loop()
{

}
```

8. Carregando imagens

O próximo exemplo mostra como carregar uma imagem proveniente do SD CARD. Tal imagem deve estar no formato bmp e ter resolução de 160x128 pixels. Para este exemplo foi considerada a imagem salva com o nome arduino.bmp.

Observe que foi declarado um novo objeto chamado imagem proveniente da classe PImage além de um pino adicional para controle de seleção do SD CARD poder ser utilizado. No bloco `setup()` a imagem é carregada usando o método image do objeto `tela` declarado anteriormente.

```
#include <SPI.h>
#include <TFT.h>
#include <SD.h>

#define SD_CS    11
#define LCD_CS   10
#define DC       9
```

```
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
PImage imagem;

void setup()
{
  while (!SD.begin(SD_CS));
  tela.begin();
  tela.background(255,255,255);
  imagem = tela.loadImage("arduino.bmp");
  if (imagem.isValid() == true)
    tela.image(imagem, 0, 0);
}

void loop()
{

}
```

Neste programa o mesmo apenas continua sua execução caso não haja erro na conexão com a memória SD CARD, identificado através do comando `while (!SD.begin(SD_CS))`. A imagem somente é carregada caso esteja na formatação correta.

Capítulo 3
Marseus van Schrieck

1. Introdução

Otto Marseus van Schrieck (Nijmegen, 1619, – 22 de junho 1678, Amsterdam) foi um pintor de paisagens neerlandês.



Fonte: https://pt.wikipedia.org/wiki/Otto_Marseus_van_Schrieck

A proposta desta experiência é apresentar a cada 30s uma imagem relacionada ao tema de uma seleção de 10 imagens. Para isso, sugere-se que o leitor salve no SD CARD dez imagens no mesmo formato do último exemplo do capítulo anterior e salve-os com os nomes img1.bmp, img2.bmp até img10.bmp.

Obs.: *Através de uma busca na internet é possível localizar as imagens relacionadas ao título da obra.*

2. Programação

Acompanhe a seguir o exemplo proposto. Verifique que boa parte do capítulo anterior foi utilizada.

```
#include <SPI.h>
#include <TFT.h>
#include <SD.h>

#define SD_CS    11
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
PImage imagem;

void setup()
{
  while (!SD.begin(SD_CS));
  tela.begin();
  tela.background(255,255,255);
```

```
}

void loop()
{
  imagem = tela.loadImage("img1.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img2.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img3.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img4.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img5.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img6.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
  imagem = tela.loadImage("img7.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img8.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img9.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img10.bmp");
  tela.image(imagem, 0, 0);
```

```
  delay(30000);
}
```

Observe que no bloco loop a cada 30s uma imagem é carregada na tela, seguindo os nomes img1.bmp até img10.bmp. Teste o programa para comprovar seu funcionamento.

Capítulo 4
Otto Stark

1. Introdução

Otto Stark (Indianapolis, 1859–1926[1]) foi um pintor impressionista americano.



Fonte: https://pt.wikipedia.org/wiki/Otto_Stark

A proposta desta experiência é apresentar a cada 30s uma imagem relacionada ao tema de uma seleção de 10 imagens. Para isso, sugere-se que o leitor salve no SD CARD dez imagens no mesmo formato do último exemplo do capítulo anterior e salve-os com os nomes img1.bmp, img2.bmp até img10.bmp.

Obs.: *Através de uma busca na internet é possível localizar as imagens relacionadas ao título da obra.*

2. Programação

Acompanhe a seguir o exemplo proposto. Verifique que boa parte do capítulo anterior foi utilizada.

```
#include <SPI.h>
#include <TFT.h>
#include <SD.h>

#define SD_CS    11
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
PImage imagem;

void setup()
{
  while (!SD.begin(SD_CS));
```

```
  tela.begin();
  tela.background(255,255,255);
}

void loop()
{
  imagem = tela.loadImage("img1.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img2.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img3.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img4.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img5.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img6.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
  imagem = tela.loadImage("img7.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img8.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img9.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
```

```
  imagem = tela.loadImage("img10.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
}
```

Observe que no bloco loop a cada 30s uma imagem é carregada na tela, seguindo os nomes img1.bmp até img10.bmp. Teste o programa para comprovar seu funcionamento.

Capítulo 5
Otto van Veen

1. Introdução

Otto van Veen (1556 – 1629) foi um pintor que se desenvolveu na Antuérpia e Bruxelas.



Fonte: https://pt.wikipedia.org/wiki/Otto_van_Veen

A proposta desta experiência é apresentar a cada 30s uma imagem relacionada ao tema de uma seleção de 10 imagens. Para isso, sugere-se que o leitor salve no SD CARD dez imagens no mesmo formato do último exemplo do capítulo anterior e salve-os com os nomes img1.bmp, img2.bmp até img10.bmp.

Obs.: *Através de uma busca na internet é possível localizar as imagens relacionadas ao título da obra.*

2. Programação

Acompanhe a seguir o exemplo proposto. Verifique que boa parte do capítulo anterior foi utilizada.

```
#include <SPI.h>
#include <TFT.h>
#include <SD.h>

#define SD_CS    11
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
PImage imagem;

void setup()
{
  while (!SD.begin(SD_CS));
  tela.begin();
  tela.background(255,255,255);
```

```
}

void loop()
{
  imagem = tela.loadImage("img1.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img2.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img3.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img4.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img5.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img6.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
  imagem = tela.loadImage("img7.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img8.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img9.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);

  imagem = tela.loadImage("img10.bmp");
  tela.image(imagem, 0, 0);
```