Vitor Amadeu Souza

Antologia

de projetos com TFT programado no Arduino em

Artes Visuais

Volume I

© 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

“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 – Abigail de Andrade **24**

04 – Arcos romanos **28**

05 – Vexilologia do Acre **32**

06 – Adam Elsheimer **34**

07 – Adriaen Isenbrandt **38**

08 – Jacobsz Cuyp **42**

09 – Aelbrecht Bouts **46**

10 – Aert de Gelder **50**

11 – Vexilologia do Afeganistão **54**

12 – Vexilologia da África do Sul **56**

13 – Agnolo Bronzino **58**

14 – Agostinho José da Mota **62**

15 – Agostino Carracci **66**

16 – Aine Bru **70**

17 – Vexilologia de Alagoas **74**

18 – Vexilologia da Albânia **76**

19 – Albert Bierstadt **78**

20 – Pensamentos de Albert Einstein **82**

21 – Albert Gottschalk **85**

22 – Albert Lebourg **89**

23 – Albrecht Altdorfer **93**

24 – Albrecht Dürer **97**

25 – Vexilologia da Alemanha **101**

26 – Alessandro Algardi **103**

27 – Alexandre Cabanel **107**

28 – Alfred East **111**

29 – Alfred Sisley **115**

30 – Pensamentos de Allan Kardec **119**

31 – Almeida Júnior **122**

32 – Alvise Vivarini **126**

33 – Vexilologia do Amapá **130**

34 – Vexilologia de Amazonas **132**

35 – Ambrogio Bergognone **134**
36 – Ambrosius Holbein **138**
37 – Vexilologia de Andorra **142**
38 – Andrea del Castagno **144**
39 – Andrea di Bartolo **148**
40 – Andrea Mantegna **152**
41 – Andrea Schiavone **156**
42 – Andrea Solario **160**
43 – Vexilologia de Angola **164**
44 – Anne-Louis Girodet **166**
45 – Vexilologia de Antígua e Barbuda **170**
46 – Antoine-Jean Gros **172**
47 – Antonello da Messina **176**
48 – Antoniazzo Romano **180**
49 – Antonio Campi **184**
50 – António de Holanda **188**
51 – António José Patrício **192**
52 – Antônio Mouro **196**
53 – Antônio Parreiras **200**
54 – Antonio di Jacopo Pollaiolo **204**
55 – Antonio Tempesta **208**
56 – Antonio Vivarini **212**
57 – Antoon van Dyck **216**
58 – Apelles de Cos **220**
59 – Vexilologia da Arábia Saudita **224**
60 – Vexilologia da Argélia **226**
61 – Vexilologia da Argentina **228**
62 – Pensamentos de Aristóteles **230**
63 – Jean-Baptiste Armand Guillaumin **233**
64 – Armas Nacionais **237**
65 – Vexilologia da Armênia **241**
66 – Arquitetura antiga **243**
67 – Arquitetura Barroca **247**
68 – Arquitetura contemporânea **251**
69 – Arquitetura gótica **255**
70 – Arquitetura medieval **259**
71 – Arquitetura idade média **263**
72 – Arquitetura Neoclássica **267**

73 – Arquitetura pré-histórica **271**
74 – Arquitetura românica **275**
75 – Arte acadêmica **279**
76 – Arte barroca **283**
77 – Arte Bizantina **287**
78 – Arte Celta **291**
79 – Arte clássica **295**
80 – Cubismo **299**
81 – Arte Egéia **303**
83 – Arte Egípcia **307**
84 – Arte Fenícia **311**
85 – Arte Gótica **315**
86 – Arte Hagiográfica **319**
87 – Impressionismo **323**
88 – Arte Islâmica **327**
89 – Maneirismo **331**
90 – Estilo Manuelista **335**
91 – Arte Mesopotâmica **339**
92 – Arte Mudéjar **342**
93 – Arte Nazarena **346**
94 – Arte Neoclássica **350**
95 – Arte Neolítica **354**
96 – Arte Paleolítica **357**
97 – Arte Persa **360**
98 – Pós-impressionismo **364**
99 – Arte Pré-românica **368**
100 – Realismo **372**
101 – Arte Renascentista **376**
102 – Rococó **380**
103 – Arte Românica **384**
105 – Romantismo **388**
106 – Arte Sassânida **392**
107 – Surrealismo **396**
108 – Arte Tumular **400**
109 – Artemisia Gentileschi **404**
110 – Pensamentos de Arthur Schopenhauer **408**
111 – Asher Brown Durand **411**
112 – Vexilologia da Austrália **415**

113 – Vexilologia da Áustria **417**
114 – Vexilologia Azerbaijão **419**
115 – Azulejo **421**
116 – Vexilologia de Bahamas **425**
117 – Vexilologia da Bahia **427**
118 – Vexilologia do Bahrein **429**
119 – Baía **431**
120 – Vexilologia de Bangladesh **434**
121 – Vexilologia de Barbados **436**
122 – Barra **438**
123 – Pensamentos de Baruch Spinoza **440**
124 – Vexilologia Belga **443**
125 – Vexilologia de Belize **445**
126 – Benedetto Ghirlandaio **447**
127 – Vexilologia de Benim **451**
128 – Benozzo Gozzoli **453**
129 – Bernardino Campi **457**
130 – Bernardino Luini **461**
131 – Bernhard Strigel **465**
132 – Bernini **469**
133 – Berthe Marie Pauline Morisot **473**
134 – Biagio d'Antonio **477**
135 – Vexilologia da Bielorrússia **481**
136 – Blaise Pascal **483**
137 – Vexilologia da Bolívia **486**
138 – Bonifazio Veronese **488**
139 – Vexilologia de Bósnia e Herzegovina **492**
140 – Vexilologia de Botswana **494**
141 – Bramantino **496**
142 – Brasão **500**

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.



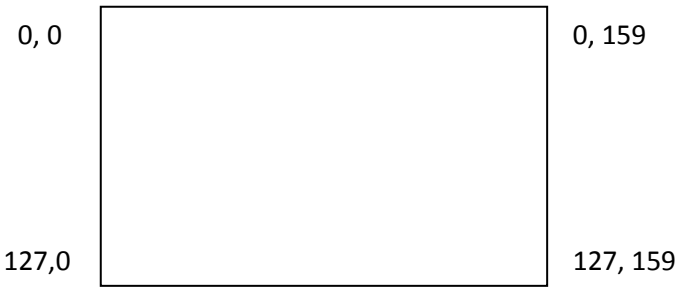
Fonte: https://www.arduino.cc/en/Guide/TFT#toc3

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.

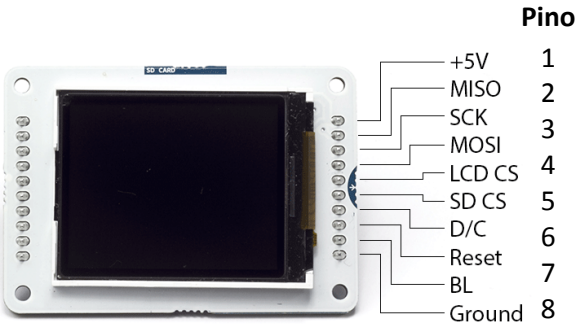
Por ter uma resolução de 160x128 pixels, podemos ver esta tela como um plano cartesiano, cujos pontos estão apresentados a seguir.



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
Abigail de Andrade

1. Introdução

Abigail de Andrade (Vassouras, 1864 — Paris, 1890) foi pintora brasileira que foi reconhecida e premiada no Salão Imperial de 1884.



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

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);

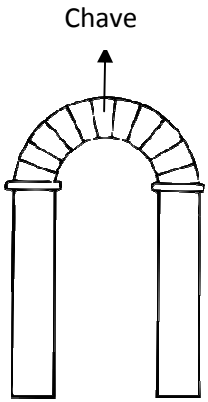
  imagem = tela.loadImage("img10.bmp");
  tela.image(imagem, 0, 0);
  delay(30000);
}
```

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
Arcos romanos

1. Introdução

Os arcos permitem maior distribuição do peso e, por conseguinte, um vão maior se comparado com o sistema de colunas utilizado pelos gregos. A peça principal do arco é a chave que é o bloco superior que trava a estrutura do arco.



A abobada segue o conceito dos arcos romanos, aumentando significativamente o espaço interno de um ambiente se comparado a arquitetura grega que faz muito uso de colunas e é usado mais para apreciação externa do que interna. O Panteão de Roma é um

resquício do antigo império romano que possui uma abobada interna com circunferência de 43,3m.

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
Vexilologia do Acre

1. Introdução

Vexilologia é o estudo das bandeiras e vexilólogo o profissional que desenha as mesmas, sendo que vexilo era o nome usado nos estandartes do exército romano.

Através do link abaixo mais informações a respeito da história e significado da bandeira com o título desta obra podem ser obtidos.

https://pt.wikipedia.org/wiki/Bandeira_do_Acre

A proposta deste exemplo é apresentar a bandeira com o estado título desta obra. Para isso, sugere-se que o leitor salve no SD CARD a imagem no mesmo formato do último exemplo do capítulo anterior com o nome img1.bmp.

Obs.: *Através de uma busca na internet é possível localiza a imagem relacionada 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);
  imagem = tela.loadImage("img1.bmp");
  tela.image(imagem, 0, 0);
}

void loop()
{

}
```

Teste o programa para comprovar seu funcionamento.

Capítulo 6
Adam Elsheimer

1. Introdução

Adam Elsheimer (18 de março de 1578 - 11 de dezembro de 1610) foi um artista alemão que viveu em Roma.



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

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 7
Adriaen Isenbrandt

1. Introdução

Adriaen Isenbrandt (c.1490, Bruges — Julho de 1551, Bruges) foi um pintor renascentista flamengo.



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

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 8
Jacobsz Cuyp

1. Introdução

Aelbert Jacobsz Cuyp (Dordrecht, 20 de outubro de 1620 — 15 de novembro de 1691) foi um pintor holandês do período barroco, conhecido pelas pinturas de paisagens campestres.



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

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 9
Aelbrecht Bouts

1. Introdução

Aelbrecht Bouts (1451-1549, Lovaina) foi pintor no estilo gótico flamengo.



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

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 10
Aert de Gelder

1. Introdução

Aert de Gelder (ou Arent, Dordrecht, Holanda 26 de outubro de 1645 - Dordrecht, 27 de agosto de 1727) foi um pintor holandês.



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

A proposta desta experiência é apresentar a cada 30s uma imagem relacionada ao tema de uma seleção de 10 imagens. Para