

## Automacao\_Residencial\_Arduino.ino - Documentação Detalhada

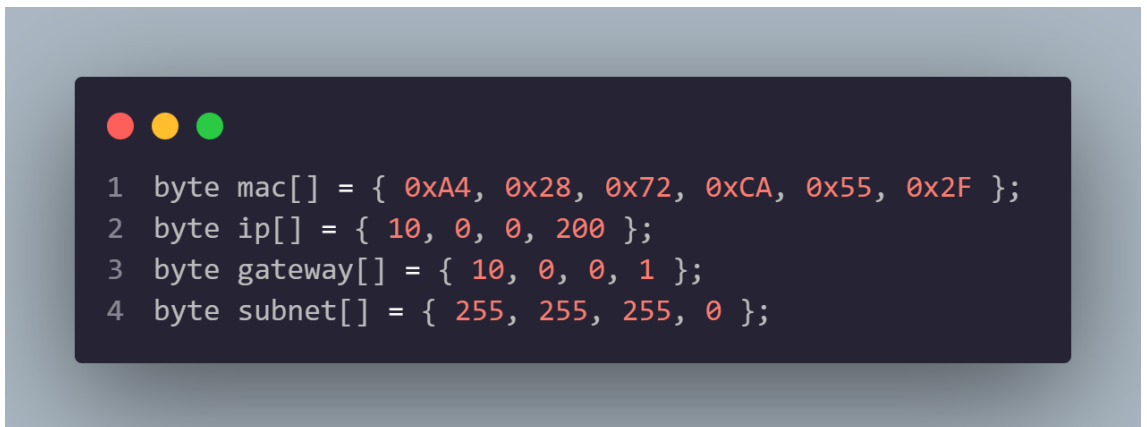
Este programa é destinado a um projeto de automação residencial utilizando Arduino e Ethernet Shield. Ele permite controlar até quatro dispositivos elétricos remotamente através de uma interface web. A documentação abaixo fornece uma explicação detalhada das principais partes do código.

---

### 1. Configurações Iniciais



- **Descrição:**
  - Inclui as bibliotecas necessárias para o funcionamento do Ethernet Shield.



#### Descrição:

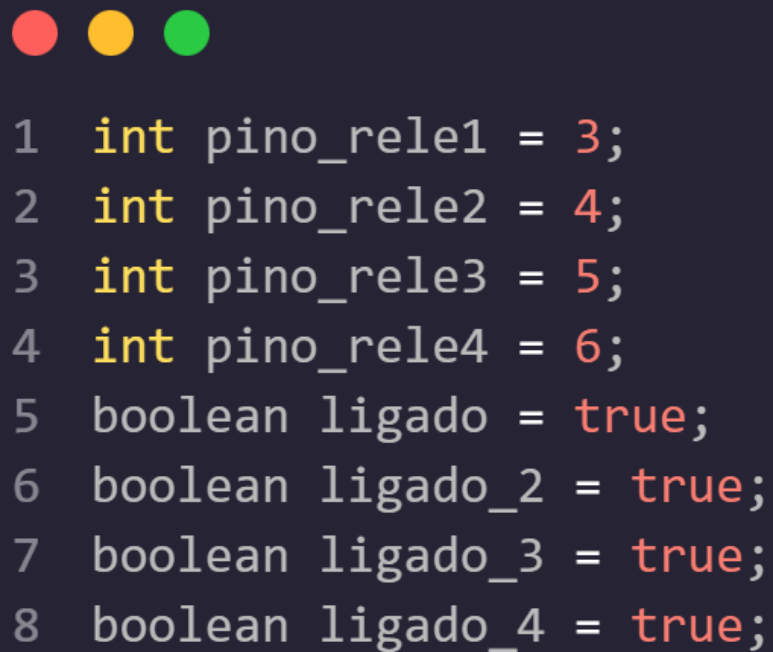
- Define as configurações de rede, como o endereço MAC e IP do Arduino.



```
1 EthernetServer server(80);
```

- **Descrição:**
  - Cria um objeto servidor para lidar com as solicitações na porta 80 (HTTP).

## 2. Configuração dos Pinos e Estados Iniciais

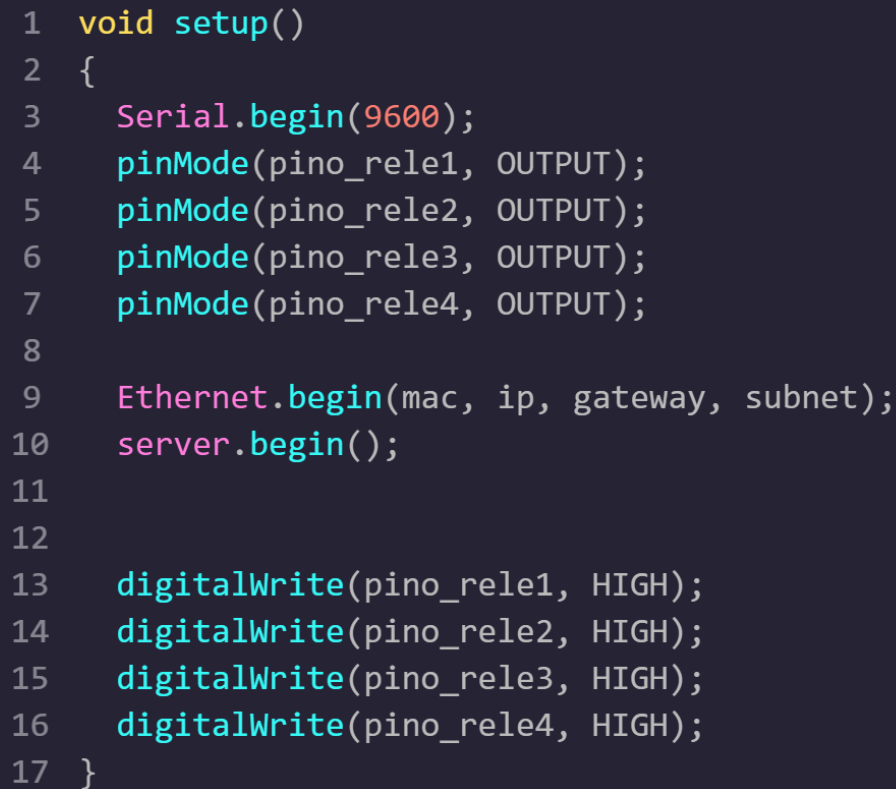


```
1 int pino_rele1 = 3;  
2 int pino_rele2 = 4;  
3 int pino_rele3 = 5;  
4 int pino_rele4 = 6;  
5 boolean ligado = true;  
6 boolean ligado_2 = true;  
7 boolean ligado_3 = true;  
8 boolean ligado_4 = true;
```

- **Descrição:**
  - Define os pinos dos relés e as variáveis de estado para cada um deles.
  - **pino\_rele1** a **pino\_rele4** representam os pinos de controle dos relés.

- **ligado, ligado\_2, ligado\_3, ligado\_4** indicam se os relés estão ligados (LOW) ou desligados (HIGH).


### 3. Configuração Inicial do Arduino



```
1 void setup()
2 {
3   Serial.begin(9600);
4   pinMode(pino_rele1, OUTPUT);
5   pinMode(pino_rele2, OUTPUT);
6   pinMode(pino_rele3, OUTPUT);
7   pinMode(pino_rele4, OUTPUT);
8
9   Ethernet.begin(mac, ip, gateway, subnet);
10  server.begin();
11
12
13  digitalWrite(pino_rele1, HIGH);
14  digitalWrite(pino_rele2, HIGH);
15  digitalWrite(pino_rele3, HIGH);
16  digitalWrite(pino_rele4, HIGH);
17 }
```

- **Descrição:**
  - Configura a porta serial para comunicação de depuração.
  - Define os pinos dos relés como saída.
  - Inicializa a conexão Ethernet e o servidor na porta 80.
  - Desliga todos os relés inicialmente.

#### 4. Loop Principal



```
1 void loop()
2 {
3   EthernetClient client = server.available();
```

- Descrição:
  - O loop principal do programa. Verifica se há um cliente (interface web) disponível para conexão.



```
1 if (client) {
2   while (client.connected())
3   {
4     if (client.available())
5     {
6
7       char c = client.read();
```

- Descrição:
  - Verifica se o cliente está conectado e se há dados disponíveis para leitura.
  - Lê caracteres recebidos do cliente.



```
1  if (readString.length() < 100) {  
2      readString += c;  
3  }
```

- Descrição:
  - Acumula os caracteres lidos em uma string (readString) se o tamanho atual da string for inferior a 100 caracteres.



```
1  if (c == 'n')  
2      {
```

- Descrição:
  - Verifica se o caractere lido é 'n', indicando o final da transmissão da requisição HTTP.

```

1 //Liga o Relé 1
2 if (readString.indexOf("?ligar") > 0)
3 {
4     digitalWrite(pino_rele1, LOW);
5     ligado = false;
6 }
7 else
8 {
9     //Desliga o Relé 1
10    if (readString.indexOf("?desligar") > 0)
11    {
12        digitalWrite(pino_rele1, HIGH);
13        ligado = true;
14    }
15 }

```

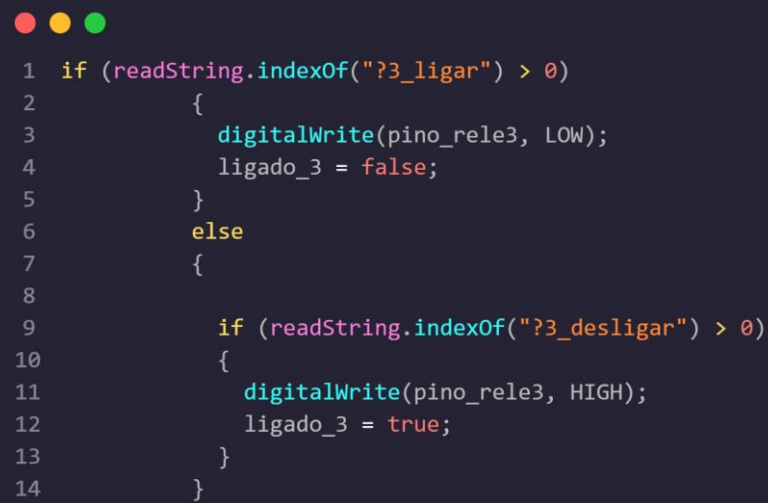
- Descrição:
  - Interpreta os comandos recebidos na string readString relacionados ao Relé 1.
  - Liga o Relé 1 se a substring "?ligar" estiver presente, desliga se "?desligar" estiver presente.
  - Atualiza a variável ligado de acordo.

```

1 if (readString.indexOf("?2_ligar") > 0)
2 {
3     digitalWrite(pino_rele2, LOW);
4     ligado_2 = false;
5 }
6 else
7 {
8
9     if (readString.indexOf("?2_desligar") > 0)
10    {
11        digitalWrite(pino_rele2, HIGH);
12        ligado_2 = true;
13    }
14 }

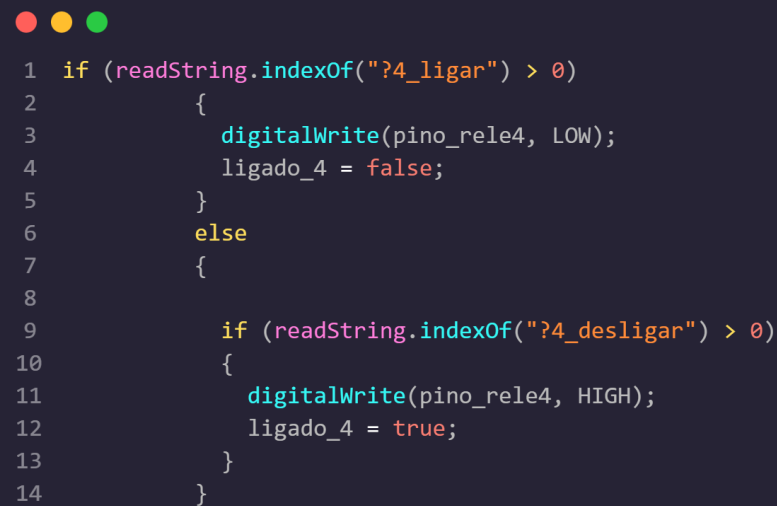
```

- Descrição:
  - Interpreta os comandos relacionados ao Relé 2 da mesma maneira descrita anteriormente.



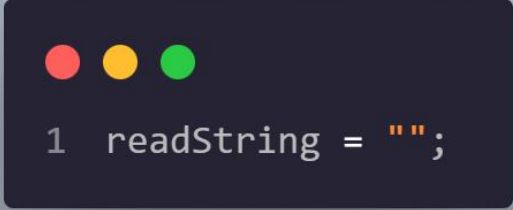
```
1  if (readString.indexOf("?3_ligar") > 0)
2      {
3          digitalWrite(pino_rele3, LOW);
4          ligado_3 = false;
5      }
6  else
7      {
8
9          if (readString.indexOf("?3_desligar") > 0)
10             {
11                 digitalWrite(pino_rele3, HIGH);
12                 ligado_3 = true;
13             }
14     }
```

- **Descrição:**
  - Interpreta os comandos relacionados ao Relé 3 da mesma maneira descrita anteriormente.



```
1  if (readString.indexOf("?4_ligar") > 0)
2      {
3          digitalWrite(pino_rele4, LOW);
4          ligado_4 = false;
5      }
6  else
7      {
8
9          if (readString.indexOf("?4_desligar") > 0)
10             {
11                 digitalWrite(pino_rele4, HIGH);
12                 ligado_4 = true;
13             }
14     }
```

- **Descrição:**
  - Interpreta os comandos relacionados ao Relé 4 da mesma maneira descrita anteriormente.



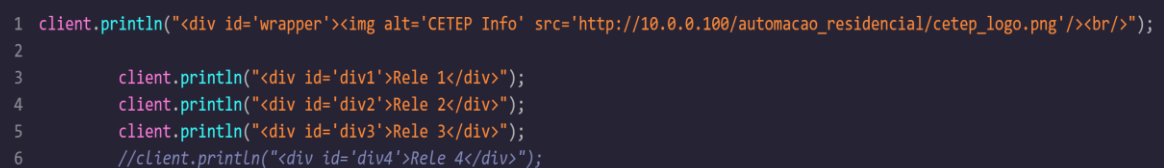
```
1 readString = "";
```

- **Descrição:**
  - **Limpa a string readString após processar os comandos.**



```
1 client.println("HTTP/1.1 200 OK");
2     client.println("Content-Type: text/html");
3     client.println();
4     client.println("<html>");
5     client.println("<head>");
6     client.println("<title>CETEP Info - Automacao Residencial</title>");
7     client.println("<meta http-equiv='Content-Type' content='text/html; charset=ISO-8859-1'>");
8     client.println("<meta name='viewport' content='width=720, initial-scale=0.5' />");
9     client.println("<link rel='stylesheet' type='text/css' href='http://10.0.0.100/automacao_residencial/automacao_residencial.css' />");
10    client.println("<script type='text/javascript' src='http://10.0.0.100/automacao_residencial/automacao_residencial.js'></script>");
11    client.println("</head>");
12    client.println("<body>");
```

- **Descrição:**
  - **Gera o cabeçalho da resposta HTTP indicando sucesso (código 200 OK) e configurações de conteúdo HTML.**



```
1 client.println("<div id='wrapper'><img alt='CETEP Info' src='http://10.0.0.100/automacao_residencial/cetep_logo.png'><br/>");
2
3     client.println("<div id='div1'>Rele 1</div>");
4     client.println("<div id='div2'>Rele 2</div>");
5     client.println("<div id='div3'>Rele 3</div>");
6     //client.println("<div id='div4'>Rele 4</div>");
```

- **Descrição:**
  - **Adiciona elementos HTML para criar a estrutura básica da interface web com três divs representando os relés.**





```
1 client.print("<div id='rele'></div><div id='estado' style='visibility: hidden;'>");
2     client.print(ligado);
3     client.println("</div>");
4     client.println("<div id='botao'></div>");
```

- **Descrição:**
  - Adiciona elementos HTML dinâmicos para apresentar o estado atual e o botão de controle do Relé 1.



```
1 client.print("<div id='rele_2'></div><div id='estado_2' style='visibility: hidden;'>");
2     client.print(ligado_2);
3     client.println("</div>");
4     client.println("<div id='botao_2'></div>");
```

- **Descrição:**
  - Adiciona elementos HTML dinâmicos para apresentar o estado atual e o botão de controle do Relé 2.



```
1 client.print("<div id='rele_3'></div><div id='estado_3' style='visibility: hidden;'>");
2     client.print(ligado_3);
3     client.println("</div>");
4     client.println("<div id='botao_3'></div>");
```

- **Descrição:**
  - Adiciona elementos HTML dinâmicos para apresentar o estado atual e o botão de controle do Relé 3.



```
1 client.println("</div>");
2
3     client.println("<script>AlterarRele1()</script>");
4     client.println("<script>AlterarRele2()</script>");
5     client.println("<script>AlterarRele3()</script>");
6     //client.println("<script>AlterarRele4()</script>");
7     client.println("</div>");
8
9     client.println("</body>");
10    client.println("</html>");
```

- **Descrição:**
  - Fecha a div principal e adiciona scripts JavaScript para atualizar din

---

Essa documentação fornece uma visão geral do código,