

arduino 上使用 ENC28J60 以太网控制器的教程—Arduino 处理图片

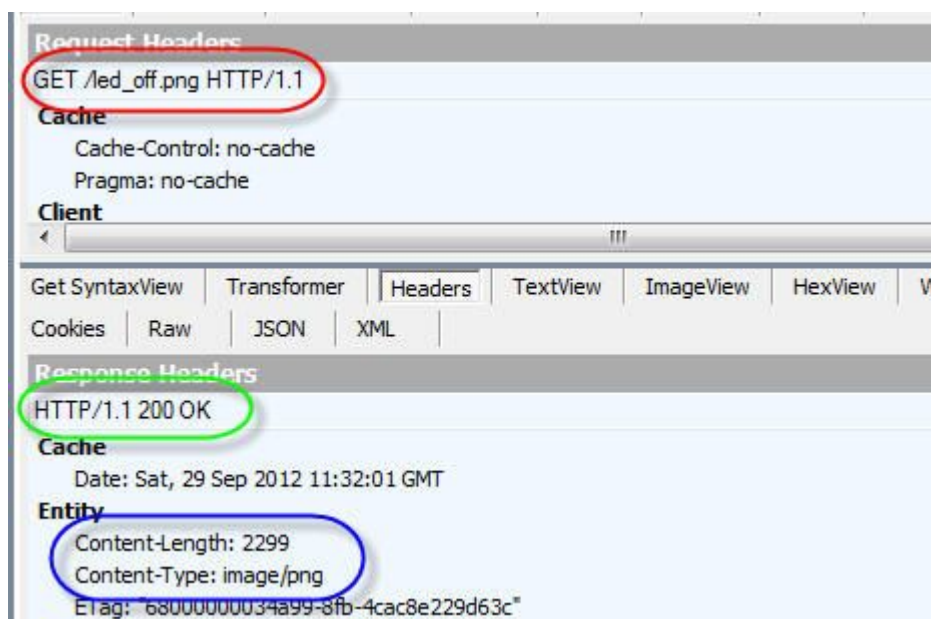
我发表上一篇教程[如何使用网页控制 Led](#)后，有人问我如何控制一个以上的 Led，今天我来告诉你如何控制两个 Led，并用更为时尚的网页元素—图像。

图像

首先你要明白当一个网页引用了一些外部资源（如图像，javascript...）后的处理流程：

- 用户的浏览器连接 web 服务器，请求 HTML 网页。
- 浏览器解析页面，找到外部资源。
- 浏览器向服务器请求每一个外部资源。

当服务器应答，它在响应的头文件中告诉浏览器他发送的文件 [MIME](#) 类型。下面这个例子（用 [Fiddler](#) 嗅探）是关于 PNG 图像。



Arduino 代码应该能够：

- 读取浏览器的请求（保存在 Ethernet:buffer）。
- 识别浏览器请求的资源（HTML 页面、图像...）。
- 创建一个正确的头文件（含类型 Content-Type）。
- 发送头文件和请求的资源到浏览器。

二进制资源

图片是一个二进制文件，在这个例子中我们应该能够把它以字节数组（byte arrays）的形式存放在我们的代码中。我发现

[bin2h](#) 这个工具可以帮助我们进行转换。

转换结果是一个文本文件：

```
led_off.txt - Notepad
File Edit Format View Help
//file auto-generated from led_off.jpg by bin2h.exe
size_t data_len = 1237;
unsigned char data[1237]=
{
    0xFF,0xD8,0xFF,0xE1,0x00,0x18,0x45,0x78,0x69,0x
    0x00,0x49,0x49,0x2A,0x00,0x08,0x00,0x00,0x00,0x
    0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xEC,0x00,0x
    0x75,0x63,0x6B,0x79,0x00,0x01,0x00,0x04,0x00,0x
```

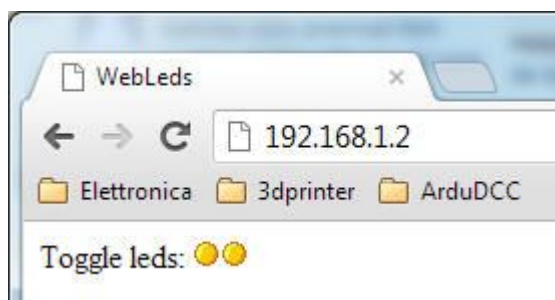
为了节省 Arduino 内存，我们用 PROGMEM 指令存储在 flash 中：

```
_8_WebLeds
#define LED2PIN 3
PROGMEM prog_char led_off[] = {
    0x89, 0x50, 0x4E, 0x47, 0x0D, 0x0A, 0x1A, 0x0A, 0
    0x00, 0x00, 0x00, 0x0E, 0x08, 0x04, 0x00, 0x00, 0
    0x42, 0x00, 0xAE, 0xCE, 0x1C, 0xE9, 0x00, 0x00, 0
    0x00, 0x00, 0x00, 0x09, 0x70, 0x48, 0x59, 0x73, 0
```

Arduino

同往常一样，完整的代码共享在 [GitHub](#)。

下面是我的代码将显示在浏览器的网页。



当用户点击其中一个图标，浏览器将请求该页面并添加**?LEDx** 后缀：Arduino 将改变相应的 Led 状态和图标的颜色。

让我们先来了解一些代码片段。

1. `if(strstr((char *)Ethernet::buffer + pos, "GET /led_off.png") != 0)`
2. `send_png_image(led_off, sizeof(led_off));`
3. `else if(strstr((char *)Ethernet::buffer + pos, "GET /led_on.png") != 0)`
4. `send_png_image(led_on, sizeof(led_on));`

我们的代码解析浏览器的请求，如果请求两个图片中的一个图片，调用 **send_png_image()** 方法，把它发送给浏览器。

1. `void send_png_image(PGM_P png_image, unsigned int image_size) {`

```

2.
3.   BufferFiller bfill = ether.tcpOffset();
4.   bfill.emit_p(PSTR("HTTP/1.0 200 OK\r\n"
5.     "Content-Type: image/png\r\n\r\n"));
6.   bfill.emit_raw_p(png_image, image_size);
7.   ether.httpServerReply(bfill.position());
8. }

```

这个方法准备正确头文件及用 **emit_raw_p** 方法添加图形文件的二进制数据到响应，最后用 **httpServerReply()** 发送响应给浏览器。

```

1.   if(strstr((char *)Ethernet::buffer + pos, "GET /?LED1") != 0) {
2.     led1Status = !led1Status;
3.     digitalWrite(LED1PIN, led1Status);
4.   }

```

如果浏览器的请求中包含**?LEDx**，Led 的状态发生改变，HTML 页面重新创建，根据 Led 的状态设定正确的图标。

```

if(led1Status) bfill.emit_p(PSTR("<a href=\"//?LED1\"><img src=\"led_on.png\"></a>"));
else bfill.emit_p(PSTR("<a href=\"//?LED1\"><img src=\"led_off.png\"></a>"));

if(led2Status) bfill.emit_p(PSTR("<a href=\"//?LED2\"><img src=\"led_on.png\"></a>"));
else bfill.emit_p(PSTR("<a href=\"//?LED2\"><img src=\"led_off.png\"></a>"));

```