**Controlando dispositivos pela Internet**

Anteriormente, aprendemos a conectar o NodeMCU a internet e a enviar uma requisição, naquele caso, para acender LED.

Nesse capítulo, iremos controlar um dispositivo pela Internet. Porém, faremos uso de alguns recursos para simplificar e tornar mais amigáveis as aplicações IoT. Um dos recursos que utilizaremos é o **jQuery**, uma biblioteca JavaScipt que tornam as páginas mais interativas e simplifica as linhas de código JS e é compatível com o AJAX, que vermos logo mais.

Também, utilizaremos o **Bootstrap** é um framework que torna as aplicações web responsivas, ou seja, o site adapta seu tamanho da tela do dispositivo que está sendo utilizado. E, o **AJAX** (Asynchronous JavaScript and XML) que fornece uma maneira de realizar requisições ao servidor que retornará dados, e não uma página HTM completa com marcadores e conteúdo.

Para esse experimento vamos precisar de...

* ESP3;
* Protoboard;
* 1 LED;
* 1 resistor 220 ou 330 Ω;
* Jumpers;

Faça a montagem a seguir...

Montagem.

**Página HTML**

Nesse modelo, criaremos uma página HTML. No cabeçalho, vamos importar as bibliotecas e o framework que citamos anteriormente (1). Em 2 e 3, no evento de clique nos botões de Acender e Apagar, utilizaremos requisições AJAX, que aumenta o desempenho e possibilita uma maior interação.

Texto

Descrição gerada automaticamente

Depois de criar, utilizando o utilitário Ampy, que possibilita a manipulação de arquivos NodeMCU, de acordo com a linha a seguir. Fique atento a porta que está sendo utilizada.



**Código**

Agora, copie o código a seguir no Thonny e execute-o.

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente com confiança média

**Explicação**

Primeiramente, importaremos o módulo network, para acessarmos recursos de rede:



Necessitaremos da biblioteca usocket, que será importada como socket:



Importaremos o módulo machine para acessar os pinos da placa:



E o módulo gc, usado para gerenciamento de memória:



Chamaremos a função gc.collect(), que coletará e realocará as informações:



Configuraremos o pino 23 como uma saída digital e chamaremos o objeto de LED:



Agora, para implementar o código-fonte do servidor HTML, criaremos a função obter\_arquivo, que obtém e retorna o conteúdo existente no sistema de Arquivos do NodeMCU:

Texto

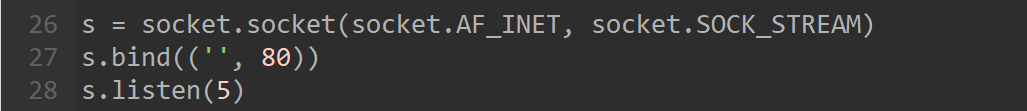
Descrição gerada automaticamente

Após isso, vamos conectar a placa a um access point:

Texto

Descrição gerada automaticamente

Criaremos um soquete, o conectaremos a porta TCP80 e limitaremos a quantia de conexões simultâneas:



A seguir, criaremos um laço que permiti pedidos de conexão, processa a requisição efetuada pelo cliente e envia o conteúdo:

Texto

Descrição gerada automaticamente

Criaremos um trecho de código para identificar qual botão está sendo pressionado, e caso nenhum botão seja acionado o led permanece apagado:

Texto

Descrição gerada automaticamente

Agora, chamaremos a função criada anteriormente para obter o conteúdo da página http-led-bootstap.html, que desenvolvemos e copiamos para o sistema de arquivos da ESP32 utilizando o AMPY:

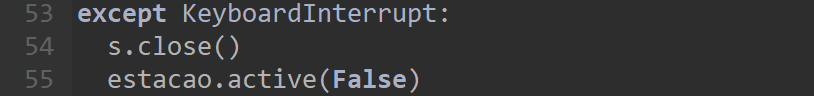


Enviaremos o cabeçalho da página e as instruções HTML são enviadas:

Texto

Descrição gerada automaticamente

Por fim, a conexão é encerrada. Entretanto, a estação continua ativa aguardando novas conexões:



**DESAFIO**

1. Assim como na última aula, substitua o led por um servo motor e, simule um portão. Um botão para abrir (90°) e um para fechar (0°).