DESENVOLVIMENTO WEB 2

Jair C Leite

EXPRESS JS

Express JS

- Versão Node.JS mínima e flexível para criar servidores HTTP
  - Utiliza o módulo http do Node.js
- Oferece recursos robustos para aplicações Web e em dispositivos móveis
- Fácil para criar APIs

Express4.16.4
Fast, unopinionated, minimalist web framework for Node.js

(expressjs.com)

Instalando Express JS

- Instale o express usando o npm
  - npm init
  - npm install express --save
- Adicionalmente (mais tarde) instale ainda:
  - body-parser – para processar JSON, Raw, Text e URL
    - npm install body-parser –save
  - cookie-parser – para Cookies
    - npm install cookie-parser –save
  - multer – para multipart/form-data.
    - npm install multer --save

Exemplo – Olá Mundo

```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
   res.send('Olá Mundo');
})

var server = app.listen(8081, function () {
   var host = server.address().address
   var port = server.address().port

   console.log("Exemplo escutando em http://%s:%s", host,
  port)
})
```

Exemplo – roteando métodos HTTP

```
var express = require('express');
var app = express();

// This responds with "Hello World" on the homepage
app.get('/', function (req, res) {
   console.log("GET request para homepage");
   res.send('Hello GET');
})

// This responds a POST request for the homepage
app.post('/', function (req, res) {
   console.log("POST request para homepage");
   res.send('Hello POST');
})

// This responds a DELETE request for the /del_user page.
app.delete('/del_user', function (req, res) {
   console.log("DELETE request para /del_user");
   res.send('Hello DELETE');
})
```

Usando recursos estáticos

- Recursos estáticos são
  - Páginas HTML, programas JS, estilos CSS, imagens,..
- O middleware express.static permite ao servidor Express tratar recursos estáticos
  - app.use(express.static('public'));
  - Onde 'public' é o diretório (pasta) onde os arquivos estáticos estarão colocados.

Crie uma pasta 'public' ou com o nome que você desejar e coloque arquivos lá

Exemplo – recursos estáticos

```
var express = require('express');
var app = express();

app.use(express.static('public'));

app.get('/', function (req, res) {
   res.send('Hello World');
})

var server = app.listen(8081, function () {
   var host = server.address().address
   var port = server.address().port

   console.log("Exemplo escutando em http://%s:%s", host, port)
})
```

Manuseando formulários – 1

- O servidor express pode trabalhar de forma tradicional para processar formulários HTML
- Seja um formulário para nome e sobrenome:

```
<html>
  <body>

    <form action = "http://127.0.0.1:8081/process_get" method =
"GET">
       Nome: <input type = "text" name = "primnome">  <br>
       Sobrenome: <input type = "text" name = "sobrenome">
       <input type = "submit" value = "Submit">
    </form>

  </body>
</html>
```

Manuseando formulários – 2

- O servidor express para processar o formulário seria

```
...
app.use(express.static('public'));
app.get('/express-form1.html', function (req, res) {
   res.sendFile( __dirname + "/" + "express-form1.html" );
})

app.get('/process_get', function (req, res) {
   // Prepare output in JSON format
   response = {
      primnome:req.query.primnome,
      sobrenome:req.query.sobrenome
   };
   console.log(response);
   res.end(JSON.stringify(response));
})
...
```

Usando POST para enviar arquivos

- Para enviar um arquivo para o servidor, use POST
  - Instale o body-parser e multer
    - npm install multer --save
    - npm install body-parser --save
- Arquivo HTML: express-form-upload.html

```
<form action = "http://127.0.0.1:8081/file_upload" method =
 "POST"
      enctype = "multipart/form-data">
      <input type="file" name="file" size="50" />
      <br />
      <input type = "submit" value = "Enviar arquivo" />
 </form>
```

express-upload.js – parte 1

```
var bodyParser = require('body-parser');
var multer  = require('multer');

app.use(express.static('public'));
app.use(bodyParser.urlencoded({ extended: false }));
app.use(multer({ dest: '/tmp/'}));

app.get('/index.htm', function (req, res) {
   res.sendFile( __dirname + "/" + "index.htm" );
})
```

express-upload.js – parte 2

```
app.post('/file_upload', function (req, res) {
   console.log(req.files.file.name);
   console.log(req.files.file.path);
   console.log(req.files.file.type);
   var file = __dirname + "/" + req.files.file.name;

   fs.readFile( req.files.file.path, function (err, data) {
      fs.writeFile(file, data, function (err) {
         if( err ) {
            console.log( err );
            } else {
               response = {
                  message:'File uploaded successfully',
                  filename:req.files.file.name
               };
            }

         console.log( response );
         res.end( JSON.stringify( response ) );
      });
   });
})
```

Referências

- https://expressjs.com
- https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm