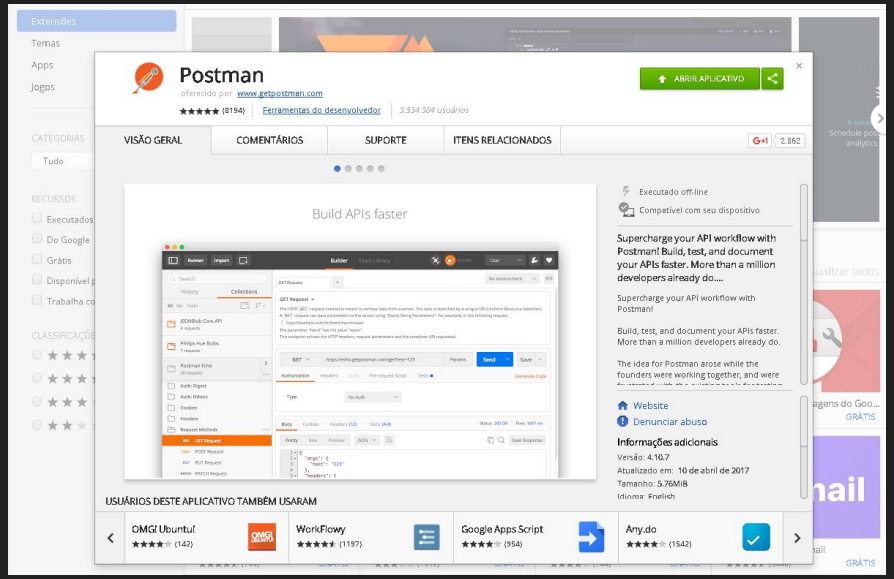
Rafael Hoffmann

UNIVALI

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Instalar Postman para enviar requisições HTTP



O Postman é uma extensão do Google Chrome muito útil para realizar testes

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Módulo HTTP

```
//hello_server.js

var http = require('http');

var server = http.createServer(function(request, response){
	response.end("<html><body>Hello World!</body></html>");
});

server.listen(3000, function(){
	console.log('Servidor Hello World rodando!');
});
```

```
//hello_server2.js
var http = require('http');

var server = http.createServer(function(request, response){
	var categoria = request.url;

	if(categoria == '/tecnologia'){
		response.end("<html><body>Noticias de tecnologia!</body></html>");
	}else
	if(categoria == '/moda'){
		response.end("<html><body>Noticias de Moda!</body></html>");
	}else
	if(categoria == '/beleza'){
		response.end("<html><body>Noticias de Beleza!</body></html>");
	}else{
		response.end("<html><body>Portal de noticias!</body></html>");
	}
});

server.listen(3000, function(){
	console.log('Servidor Hello World rodando!');
});
```

```
//hello_server3.js

var http = require('http');

var contatos = [
  {id: 1, nome: "Bruno", telefone: "9999-2222", data: new Date()},
  {id: 2, nome: "Sandra", telefone: "9999-3333", data: new Date()},
  {id: 3, nome: "Mariana", telefone: "9999-9999", data: new Date()}
];

var server = http.createServer(function(request, response){
      response.end(JSON.stringify(contatos));
});

server.listen(3000, function(){
      console.log('Servidor Hello World rodando!');
});
```

Desafio 1

1 - No módulo carros, obtenha a lista de carros diretamente do arquivo carros.csv, utilizando a função readFile do módulo fs, criando o array de caros a partir do arquivo lido. Dica: utilizar o método split() - String

2 - Crie a pasta http, juntamente com o módulo carrosHttp, utilizando o módulo http para listar os carros pela web. Dica:

```
carros.forEach(function (carro) {
        res.write('<h2>' + 'Modelo: ' + carro.modelo + '</h2>');
        res.write('<h5>' + ano: ' + carro.ano + '</h5>');
        res.write('<h5>' + cor: ' + carro.cor + '</h5>');
});
res.end();
```

Desafio 1

3 - Mude o módulo index para que seja possível rodar por meio do modo teclado ou de http, passando por parâmetro a opção -http, permitindo com que seja possível acessar a lista de carros pelo navegador na url http://localhost:3000

Para rodar no modo http, utilize o comando:

node index.js -http

Iniciando com o Express

Por que utilizá-lo?

Utilizar o módulo HTTP nativo é muito trabalhoso, aumentando a complexidade do projeto e dificultando futuras manutenções.

Express trabalha com perfeição manipulando views, routes e controllers

Focado em alta performance

Roteamento robusto

RESTFull (get, post, put, delete)

Site oficial: (http://expressjs.com) .

```
//ex01.js

var express = require('express'); //retorno da funcao
var app = express(); // executando a funcao

app.get('/', function(req,res){
	res.send("<html><body>Portal de noticias</body></html>")
});

app.get('/tecnologia', function(req,res){
	res.send("<html><body>Noticas de tecnologia</body></html>")
});

app.listen(3000, function(req, res){
	console.log("Servidor rodando com Express");
});
```

```
//ex02.js

var express = require('express'); //retorno da funcao
var app = express(); // executando a funcao

var contatos = [
  {id: 1, nome: "Bruno", telefone: "9999-2222", data: new Date()},
  {id: 2, nome: "Sandra", telefone: "9999-3333", data: new Date()},
  {id: 3, nome: "Mariana", telefone: "9999-9999", data: new Date()}
];
app.get('/', function(req,res){
	res.json(contatos);
});

app.listen(3000, function(req, res){
	console.log("Servidor rodando com Express");
});
```

```
//ex03.js

var express = require('express')
var server = express()

server.get('/', function(req, res) {
  res.send('<h1>Index!</h1>')
})

server.all('/teste', function(req, res) {
  res.send('<h1>Teste!</h1>')
})

server.listen(3000, function (){
      console.log('Executando porta 3000')
});
```

Desafio 2

1 - Modifique o módulo carrosHttp para receber parâmetros pela url e realizar a busca pelo modelo. exemplo: 127.0.0.1:3000\Gol

2 - Instale o Express utilizando o NPM: npm install express

3 - Defina uma rota utilizando o método get (express) chamada /carros, retornando um JSON .

4 - Modifique o módulo index para rodar em modo api (express) com o parâmetro -api. exemplo:

node index.js -api

Mais alguns exemplos

Desafio 3

1 - Criar lista dinâmica de contatos utilizando os métodos:

- get('/contatos') -> para retornar a lista de contatos
- get('/contatos/:id') -> retorna um contato específico
- post('/contatos') -> adicionar um novo contato na lista. Exemplo:

  contato = {‘id’: 1, ‘nome’:’Rafael’, ‘email’:raelhoff@gmail.com’}
- put('/contatos/:id') -> atualizar cadastro. Exemplo:

  contato = {‘id’: 1, ‘nome’:’Rafael Wilmar’, ‘email’:raelhoff@hotmail.com’}
- delete('/contatos/') -> delete a primeira posicao do cadastro cadastro. Dica, utilize a função

Modificando Estrutura do projeto (MVC)

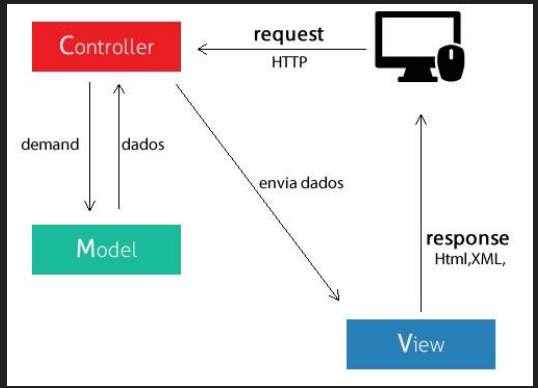
controllers -> controladores chamados pelas rotas da aplicação

models -> models que representam o domínio do problema

routes -> rotas da aplicação

views -> views do template engine

config -> configuração do express, banco de dados etc.

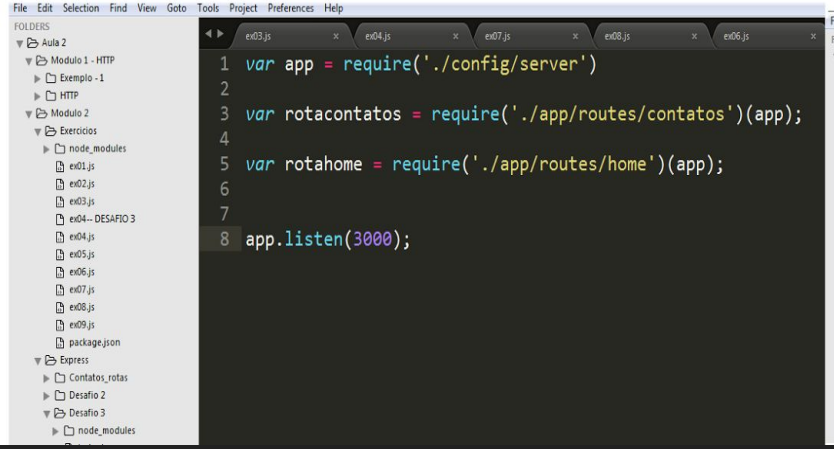


EJS (Embedded JavaScript Templates)

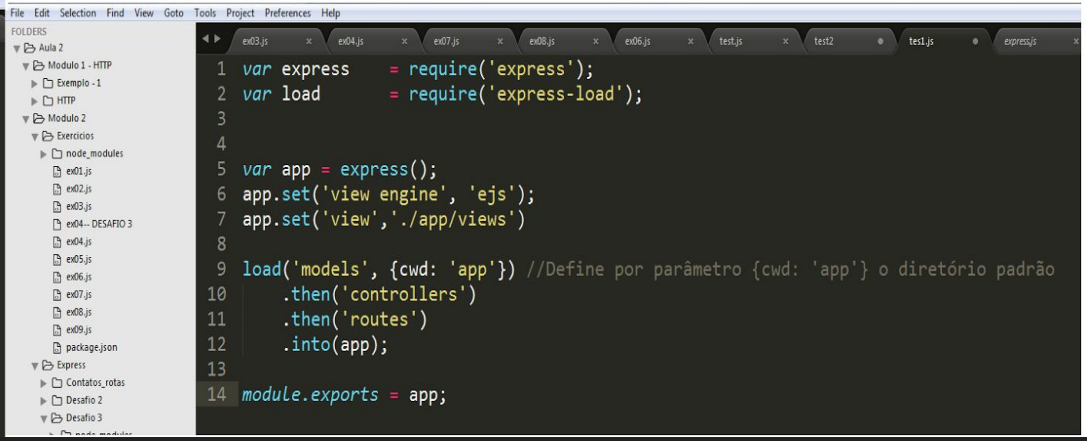
EJS: é um motor de modelos JavaScript que permite criar de forma dinâmica páginas HTML com JavaScript.

Express-load

Sem express-load



Com o uso do express-load



Refactoring projeto contatos com express-load

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Request npm

Como efetuar request (get,post, put , delete) por meio do servidor?

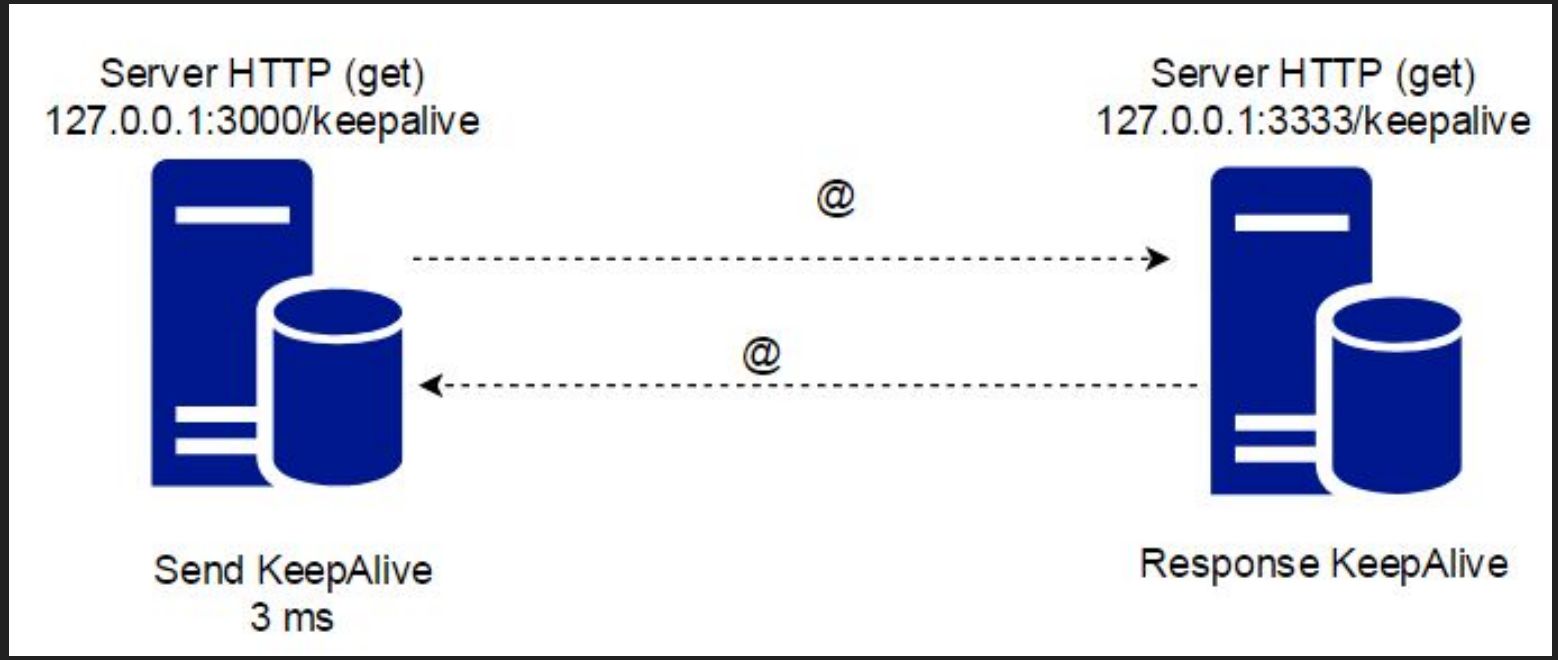
Podemos utilizar o framework request

Para instalado execute npm install request

Documentação: https://www.npmjs.com/package/request

Vamos ver um exemplo simples.

Desafio 4


Server HTTP (get)
127.0.0.1:3000/keepalive
Server HTTP (get)
127.0.0.1:3333/keepalive
@
@
Send KeepAlive
3 ms
Response KeepAlive

Introdução ao MongoDB

MongoDB é um banco de dados NoSQL

Utiliza Javascript como interface para manipulação de dados e a persistência dos dados é feita através de objetos JSON ou XML.

Nele trabalhamos com o conceito schema-less, ou seja, não existem relacionamentos de tabelas, nem chaves primárias

Site oficial: http://mongodb.com

UNIVALI
Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

MongoDB

```
//exibir todos os bancos
show dbs

// criar banco
use db_node

// verificar em qual banco voce esta
db
show dbs

// cria coleção/tabela
db.createCollection('aula')
show dbs
show collections
..
```

MongoDB

```
cls
show dbs
use db_node
show collections

//inserir informacoes (formato:json)
db.aluno.save({name:"Rafael", nota: 8, prova: 1})

var data1 = {"dia":1, "mes":5, "ano":2017}
var data2 = {"dia":2, "mes":5, "ano":2017}
var data3 = {"dia":3, "mes":5, "ano":2017}

db.data.save(data1)
db.data.save(data2)
db.data.save(data3)

show collections
show dbs
```

MongoDB

```
cls
use db_node

db.data.find()
cls

db.data.findOne()
db.data.findOne({dia: 2})
db.data.find({ano:2017})
cls
```

MongoDB

```
cls
db.data.update({
  {dia: 1}, {dia:27,mes:12,ano:2018}
```

MongoDB

```
cls
db.data.count()

db.data.remove({mes:12})
db.data.count()

db.data.remove({dia:2})
db.data.count()

db.dropDatabase()
show dbs
```

Desafio 5 - Atividades para fixação do conteúdo

1) Crie um banco de dados chamado biblioteca
2) Crie uma coleção (collection) chamada livros
3) Crie os seguintes documentos:

    titulo: Introdução a linguagem de marcação HTML

    valor: 25

..

..

..

mongoose

Mongoose

Mongoose é uma biblioteca do Nodejs que proporciona uma solução baseada em esquemas para modelar os dados da sua aplicação. Ele possui sistema de conversão de tipos, validação e criação de consultas.

Para instalar o Mongoose digite:

```
npm install Mongoose
```

Site oficial: http://mongoosejs.com/

Mongoose

Criar, Recuperar, Atualizar e Deletar (CRUD)

UNIVALI
Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Desafio 6

Refactoring projeto contatos com mongoose

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Site oficial: https://socket.io/

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Socket.io

O Socket.IO oferece uma API de JavaScript simples, baseada em eventos que te permite comunicar entre o servidor e o cliente Web em tempo real.

- É perfeito para coisas como salas de bate-papo e feeds do twitter.

Socket.IO é escrito em JavaScript tanto para front end, quanto para back-end, por isso foi projetada para rodar em um servidor Node.js.

Socket.IO usará detecção de recurso para decidir se a conexão será estabelecida com WebSocket, AJAX, Flash, etc...

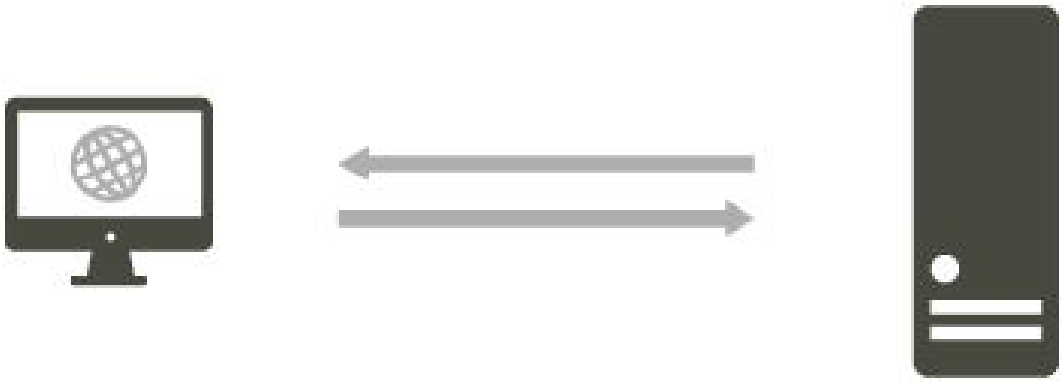
Socket.io - Cliente (index.html)

O Socket.IO está disponível para download no GitHub. Você pode incluir o socket.io.js no arquivo HTML:

```
<script src="/socket.io/socket.io.js"></script>
<script>
  var socket = io('http://localhost');
  socket.on('news', function (data) {
    console.log(data);
    socket.emit('my other event', { my: 'data' });
  });
</script>
```

Socket.io

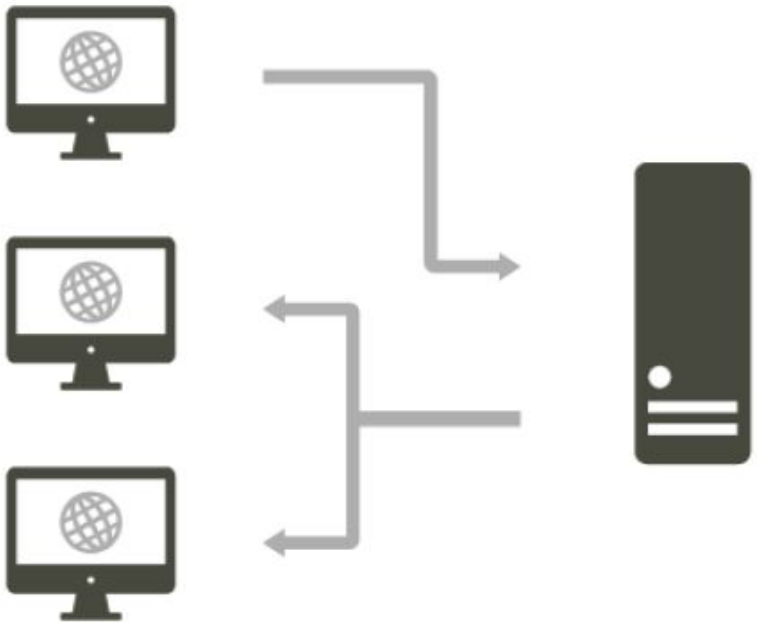
```
socket.emit(”mensagem”, “Olá!”);
```



Envia mensagens para o cliente ou servidor.

Socket.io

```
socket.broadcast.emit("mensagem", "Olá!");
```



Envia mensagens para todos os clientes, exceto o próprio emissor.

Socket.io

Exemplo 1

Socket.io - Aplicativo de bate-papo

Neste guia, criaremos um aplicativo de bate-papo básico.

Socket.io - Desafio final

- Transmita uma mensagem aos usuários conectados quando alguém se conecta ou desconecta
- Não envie a mesma mensagem para o usuário que enviou.
- Mostrar quem está online

Referências

jQuery: https://www.w3schools.com/jquery/

HTML: https://www.w3schools.com/html/html5_intro.asp

Socket.io: https://socket.io/demos/chat/