Aula Node

# O que são rotas ?

Rotas são URLs específicas que funcionam como gatilho para certas ações, elas podem ser ***estáticas*** ou ***dinâmicas***. As URLs estáticas geralmente são links no site, e as dinâmicas, geralmente, estão relacionados a um campo de pesquisa.

Junto com o “gatilho”, também é enviado um verbo HTTP, os principais são: *get, post, put, patch e delete*.

* **Get**: Quando uma ***requisição*** (request) *GET* ocorre, o servidor deve **deve** **responder** (response) apenas com um recurso / informação, esse é um método HTTP cujo único objetivo é a leitura (readonly), ou seja, receber uma informação. Ele é o método HTML mais comum da web, sempre que você digita alguma coisa na URL e pressiona ENTER, você na realidade está realizando uma requisição **get** ao servidor em questão e ele responde com algum dado, pode ser um erro, arquivo (HTML / JSON / XML), simplesmente nada etc.
* **Post**: A ***requisição*** *POST* está associada à criação de um novo recurso/informação/data. Em outras palavras, o método **post** é utilizado quando você quer **salvar** alguma coisa no servidor ou em um banco de dados. Esse verbo/método é definido através de JavaScript ou pela tag html ***form***, em seu atributo *method*. No corpo da requisição é enviado também os dados a serem salvos:



* **Put:** O método *PUT* é utilizado por suas “habilidades de atualização”. Quando você quer atualizar por **inteiro** um determinado recurso/informação/data já existente, por exemplo: realizar a substituição de um arquivo inteiro. Através desse método, a API também pode decidir criar o recurso/informação/data em questão **caso** ele **não** exista.
* **Patch:** Esse método é utilizado por suas “habilidades modificadoras”. Ele é usado quando você quer modificar parte de uma informação já existente, utilizado em sua maioria para modificar algum dado no banco de dados, por exemplo: atualizar/modificar o nome de um usuário em um arquivo JSON. No corpo de requisição também vai os novos dados.
* **Delete:** Esse é bem fácil, como o nome diz, esse método é utilizado para deletar algo, um usuário com uma id em específico, por exemplo.

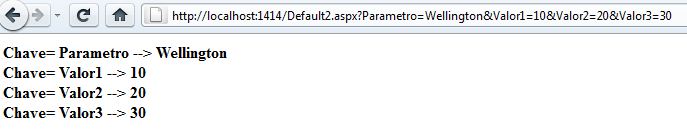
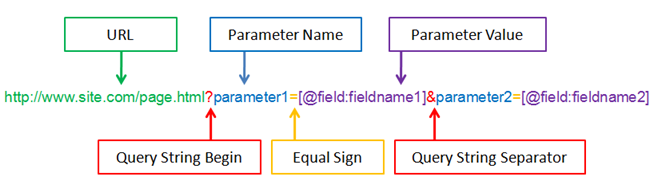
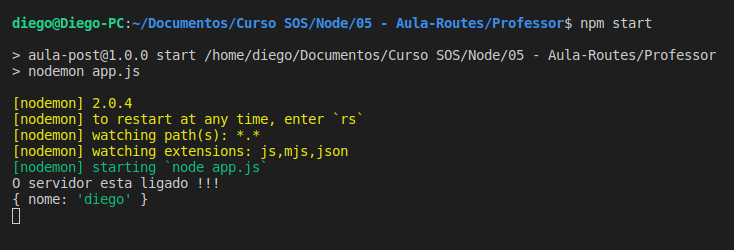
Esses métodos/verbos são processados no lado do servidor, no nosso caso, através do Node, e também são utilizados para a construção de APIs.

Leia mais em:

* <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods> (Português)
* <https://www.restapitutorial.com/lessons/httpmethods.html> (Inglês)
* <https://restfulapi.net/http-methods/> (Inglês)

# Parâmetros

Um parâmetro é um valor dinâmico, opcional, que pode ser passado através de uma rota. Esse parâmetro pode ser **resgatado** no ***Back-End*** através do objeto da requisição. Eles podem ser criado de duas formas:

* ***Query String***: Esse parâmetro é passado através da URL, utilizando o *ponto de interrogação*.
  + ***Estrutura***: No final da URL, adicione um *ponto de interrogação*( ? ), seguido por um nome e um sinal de igual ( = ) e um valor. O nome será a chave do parâmetro e o valor da chave será o que vier seguido depois do sinal de igual. Para adicionar múltiplos parâmetros, separe-os com o símbolo de “E” comercial: &. Veja alguns exemplos:
  + ***Como resgatar ?***: Ao receber a requisição através de nossa rota, podemos utilizar da propriedade ***query*** dentro do objeto da requisição, essa propriedade já vem como padrão, ela pode ser vazia ou não.
* **Route Params:** Assim como a query string, esse tbm é opcional, é enviado pela url mas não utiliza a sintaxe de interrogação e E comercial, ela é simplesmente adicionada na URL. Diferente da query string, essa precisa ser previamente declara antes no back end.
* **Body Params:** Geralmente proveniente de formulários.

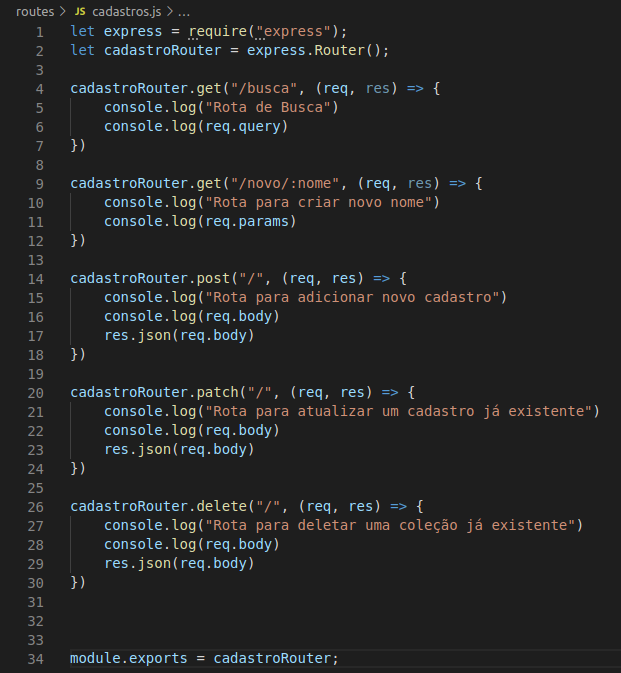
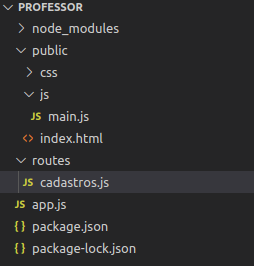
# CRUD (Create, Read, Update and Delete)

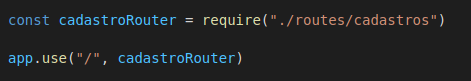
Se você construir uma API com as capacidades de ***criar***, ***ler***, ***atualizar*** e ***deletar*** ela é dita como um CRUD, uma sigla para *Create, Read, Update e Delete*. Não é explicitamente necessário utilizar os verbos HTTP para chegar à essa finalidade, apesar de ser uma boa prática. Veja na tabela a seguir as correspondências dos verbos HTTP com o CRUD:

|  |  |  |
| --- | --- | --- |
| Verbo HTTP | CRUD | Exemplo |
| Get | Read (Ler) | Lista de clientes, acessar uma página, busca |
| Post | Create (Criar) | Cadastro de um novo cliente |
| Put | Update/*Replace* (Atualizar/Substituir) | Substituir uma coleção por **inteira** |
| Patch | Update/*Modify* (Atualizar/Modificar) | Usuário realizou uma alteração no e-mail, senha, etc... |
| Delete | Delete (Deletar) | Excluir um usuário |

# Manipuladores de Rotas

Use a classe express.Router para criar manipuladores de rota modulares e montáveis. Uma instância de Router é um middleware e sistema de roteamento completo; por essa razão, ela é frequentemente referida como um “mini-aplicativo”

O seguinte exemplo cria um roteador como um módulo, define algumas rotas, e monta o módulo router em um caminho no aplicativo principal.



Fontes:

<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods>

https://restfulapi.net/http-methods/

https://expressjs.com/pt-br/guide/routing.html