

O que é REST?

Transcrição

Tudo certo para a listagem de restaurantes. Mas será que o app AluraFood se resume a listar restaurantes? Provavelmente não, dado que o usuário efetua pedidos, um restaurante tem cardápio que poderia sofrer alterações e por aí vai.

Algumas funcionalidades específicas aos responsáveis de um restaurante podem ser necessárias. E para isso o `webservice` deveria estar preparado também para lidar com essa necessidade:

Listagem de todos os restaurantes --> GET - `/restaurante`

Adicionar um restaurante --> POST - `/restaurante`

Perceba que no exemplo fictício as duas primeiras URIs são idênticas e a funcionalidade muda completamente a partir do método HTTP usado:

GET -> Listagem

POST -> Criação

A atualização poderia ter um outro endpoint como por exemplo:

PUT/PATCH - `/restaurante/1`

O número 1 ao final da URI indica um identificador a um restaurante específico.

A remoção de um restaurante poderia seguir o mesmo modelo:

DELETE - `/restaurante/1`

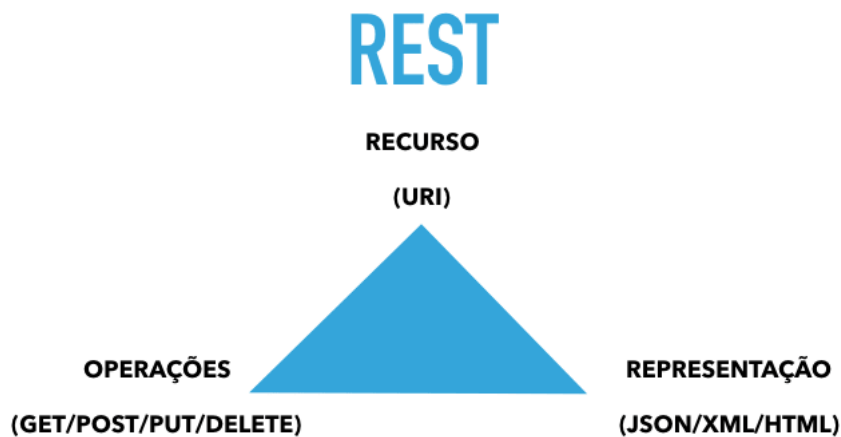
Para a busca do cardápio de um restaurante específico o endpoint gerado poderia seguir o modelo:

GET - `/restaurante/1/cardapio`

O padrão REST

Logo podemos perceber que o padrão usado pela equipe do `webservice` define que uma requisição web tem três tipos de componentes importantes: recursos (URI), operações (GET, POST, PUT, DELETE/...) e representação de dados (XML, JSON, ...).

Esses três componentes em conjuntos seguindo algumas práticas são a base para o modelo arquitetural **REST(Representational State Transfer)** ou em português **Transferência de Estado Representacional**.



Recurso

Ao criar as URIs do nosso sistema devemos levar em conta que elas representam recursos, não ações.

Em sistemas REST, nossas URIs devem conter apenas substantivos, que são nossos recursos: `/restaurante/adiciona` não é uma boa URI, pois contém um verbo e não está identificando um recurso, mas sim uma operação.

Para representar a adição de um restaurante podemos usar a URI `/restaurante` com o método HTTP POST, que representa que estamos adicionando alguma informação no sistema.

Operações

O protocolo HTTP possui operações através de métodos como: GET, POST, PUT e DELETE.

Cada método tem uma semântica diferente e juntando o método à URI deveríamos conseguir representar todas as ações do nosso sistema.

As semânticas principais são:

- GET - recupera informações sobre o recurso identificado pela URI. Ex: listar restaurante, visualizar o restaurante 1. Uma requisição GET não deve modificar nenhum recurso do seu sistema, ou seja, não deve ter nenhum efeito colateral, você apenas recupera informações do sistema.
- POST - adiciona informações usando o recurso da URI passada. Ex: adicionar um restaurante. Pode adicionar informações a um recurso ou criar um novo recurso.
- PUT - adiciona (ou modifica) um recurso na URI passada. Ex: atualizar um restaurante.
- DELETE - remove o recurso representado pela URI passada. Ex: remover um restaurante.

Representação

Quando fazemos uma aplicação não trafegamos um recurso pela rede, apenas uma representação dele. E essa representação pode ser feita de diferentes formas como JSON, XML ou HTML.

Conclusão

Nossas URIs devem representar recursos, as operações no recurso devem ser indicadas pelos métodos HTTP e podemos falar qual é o formato em que conversamos com o servidor com o `Content-Type` e `Accept` que são cabeçalhos do HTTP.