

Documentação do arquivo AccountRestController.cls

Introdução

O arquivo AccountRestController.cls define uma classe global que implementa um recurso REST para gerenciar registros da entidade Account no Salesforce. Ele utiliza a anotação @RestResource para mapear URLs e fornece métodos para realizar operações CRUD (Create, Read, Update, Delete) usando verbos HTTP apropriados.

Descrição

A classe AccountRestController é projetada para expor endpoints REST que permitem interagir com registros da entidade Account. Ela segue boas práticas de desenvolvimento REST, como validação de dados, uso de verbos HTTP adequados e gerenciamento de exceções. A classe utiliza o objeto RestContext para acessar informações de solicitação e implementa métodos para criar, recuperar, atualizar e excluir registros.

Estrutura

A classe possui os seguintes métodos:

- `getAccountById`: Recupera um registro de Account com base no ID fornecido na URL.
- `createAccount`: Cria um novo registro de Account com os dados fornecidos.
- `updateAccount`: Atualiza um registro de Account existente com base no ID fornecido na URL e nos parâmetros da solicitação.
- `deleteAccount`: Exclui um registro de Account com base no ID fornecido na URL.

Imports

Não há importações explícitas no código, pois ele utiliza recursos nativos do Salesforce.

Variáveis

- `RestRequest req`: Representa a solicitação REST atual.
- `String accountId`: Armazena o ID do registro de Account extraído da URL.
- `Account acc`: Representa o registro de Account que está sendo manipulado.

Dependências

- Objeto `RestContext`: Utilizado para acessar informações da solicitação REST.
- SOQL (Salesforce Object Query Language): Utilizado para consultar registros de Account.
- Entidade Account: Representa os registros que estão sendo manipulados.

Métodos

getAccountById

Descrição: Recupera um registro de Account com base no ID fornecido na URL.

HTTP Verbo: GET

Retorno: Um objeto Account.

Exemplo de uso:

GET /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX

createAccount

Descrição: Cria um novo registro de Account com os dados fornecidos.

HTTP Verbo: POST

Parâmetros:

- name: Nome da conta.
 - phone: Telefone da conta.
 - website: Website da conta.
- Retorno:** Uma mensagem indicando o ID do registro criado.
- Exemplo de uso:**

```
POST /services/apexrest/accounts
Body: { "name": "Empresa X", "phone": "123456789", "website": "www.empresax.com" }
```

updateAccount

Descrição: Atualiza um registro de Account existente com base no ID fornecido na URL e nos parâmetros da solicitação.

HTTP Verbo: PUT

Parâmetros:

- name: Novo nome da conta.
 - phone: Novo telefone da conta.
 - website: Novo website da conta.
- Retorno:** Uma mensagem indicando o ID do registro atualizado.
- Exemplo de uso:**

```
PUT /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX
Body: { "name": "Empresa Y", "phone": "987654321", "website": "www.empresay.com" }
```

deleteAccount

Descrição: Exclui um registro de Account com base no ID fornecido na URL.

HTTP Verbo: DELETE

Retorno: Uma mensagem indicando o ID do registro excluído.

Exemplo de uso:

```
DELETE /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX
```

Exemplo

Criar uma conta

```
POST /services/apexrest/accounts
Body: { "name": "Empresa X", "phone": "123456789", "website": "www.empresax.com" }
```

Recuperar uma conta

```
GET /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX
```

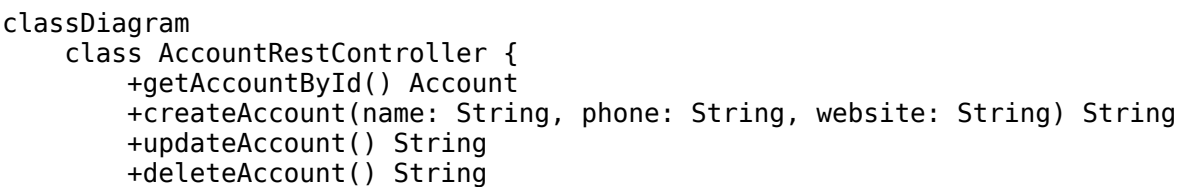
Atualizar uma conta

```
PUT /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX
Body: { "name": "Empresa Y", "phone": "987654321", "website": "www.empresay.com" }
```

Excluir uma conta

```
DELETE /services/apexrest/accounts/001XXXXXXXXXXXXXXXXX
```

Diagrama de Dependências



```
}
class RestContext {
    +request: RestRequest
}
class RestRequest {
    +requestURI: String
    +params: Map<String, String>
}
class Account {
    +Id: String
    +Name: String
    +Phone: String
    +Website: String
}
AccountRestController --> RestContext
RestContext --> RestRequest
AccountRestController --> Account
```

Notas

- A classe utiliza a cláusula LIMIT nas consultas SOQL para evitar a busca de grandes conjuntos de dados e respeitar os limites de consulta do Salesforce.
- A anotação `global` permite que a classe seja acessada por todas as fontes, incluindo chamadas externas.

Vulnerabilidades

- **Injeção SOQL:** Embora o código utilize variáveis vinculadas para evitar injeção SOQL, é importante validar os dados de entrada para garantir segurança adicional.
- **Exposição de dados sensíveis:** Certifique-se de que os dados retornados não contenham informações sensíveis que não devem ser expostas via API REST.
- **Gerenciamento de exceções:** O código não inclui tratamento explícito de exceções. É recomendável adicionar blocos try-catch para lidar com erros inesperados.