

# Documentum REST Extensibility Tutorial (5): Update and Delete Object

---

**Previous:** [Documentum REST Extensibility Tutorial \(4\): Create Object](#)

## Overview

This document is for the continuous series for Documentum REST Extensibility tutorials. Previously, we learnt how to create an alias set in the sample project. In this document, we will show codes to update and delete alias set objects.

As [Documentum REST Extensibility Tutorial \(3\): Explore The Sample Project](#) showed, there are two custom resources in the sample project.

- alias sets collection resource
- alias set object resource

We are going to implement the update and removal of any single alias set object in this tutorial. Since an alias set object in the REST API is identified by resource URI, we will follow the standard HTTP methods to update and remove the alias set object by its URI.

For the update, there are several HTTP method candidates, **PUT**, **POST** and **PATCH**. According to [RFC 7231 - Hypertext Transfer Protocol \(HTTP/1.1\): Semantics and Content](#),

- PUT can be used to create or update server state. It's idempotent, meaning that the client can send the same request repeatedly and the result state (e.g. any object attributes) won't be changed for repeated requests.
- POST can be used to create or update server state, too. It's not idempotent. The main difference against PUT is the server can have its own semantics to handle the request.
- PATCH is introduced in [RFC 5789 - PATCH Method for HTTP](#) to do partial update. But it's not as common as the other two.

Considering the above definitions, we highly recommend to use **POST method to make partial update on Documentum resources**.

For the removal, it is natural to use HTTP **DELETE** method to remove the alias set object.

Now, please open the existing *acme-rest* project at `<SDK_ROOT>/maven-kit/generated/acme-rest/`. You can use an Java IDE to open the project as a Maven project or open files in the text editor directly.

## Implementation

### Update

Open the Java class *AliasSetController.java*, add a new method within the class entity. Below is the complete Java method for the alias set update. We will explain every piece of the code.

```
@RequestMapping(    method = RequestMethod.POST,    produces = {        SupportedMediaTypes.APPLICATION_VND
```

Line 01 ~ 09 :

We annotate the controller method with Spring annotation *@RequestMapping*. HTTP method and supported media types should be specified in the request mapping.

Line 10:

We annotate the controller method with Spring annotation *@ResponseBody*. This annotation makes the method return type *AliasSet* serializable as XML or JSON message.

Line 11 ~ 17:

Here we declare some method parameters with annotations like `@PathVariable`, `@RequestBody`, `@TypedParam` and `@RequestUri`. They have been explained in [Documentum REST Extensibility Tutorial \(4\): Create Object](#).

Line 18, 25 ~ 32:

Validate the alias set object ID from the path variable, to make sure it is existed.

Line 19 ~ 20:

Call `SysObjectManager` to update the alias set object.

Line 21:

Render the model instance as the output. This will call the parent method to load the annotated resource view (`AliasSetView`) dynamically to build links.

## Delete

On the Java class `AliasSetController.java`, add a new method within the class entity. Below is the complete Java method for the alias set delete.

```
@RequestMapping(method = RequestMethod.DELETE) @ResponseStatus(HttpStatus.NO_CONTENT) public void deleteAliasSet(
```

Line 01:

We annotate the controller method with Spring annotation `@RequestMapping`. HTTP method DELETE is specified in the request mapping.

Line 02:

We annotate the controller method with Spring annotation `@ResponseStatus(HttpStatus.NO_CONTENT)`. This annotation makes the response status as 204.

Line 03 ~ 07:

Here we declare some method parameters with annotations like `@PathVariable` and `@RequestUri`. They have been explained in [Documentum REST Extensibility Tutorial \(4\): Create Object](#).

Line 08:

Validate the alias set object ID from the path variable, to make sure it is existed.

Line 09:

Call `SysObjectManager` to delete the alias set object.

Deploy and Test

Run Maven goal `mvn clean install` to rebuild the project and redeploy it to the web container again.

Open a REST client debugging tool like PostMan or DHC to test the POST and DELETE method.

***Request - update alias set:***

POST http://localhost:8080/acme-rest/repositories/REPO/alias-sets/6600000580000515 HTTP/1.1 Authorization: Basic ZG1hZG1p

***Response - update alias set:***

HTTP/1.1 200 OK Content-Type: application/vnd.emc.documentum+json;charset=UTF-8 Content-Length: 725 { "type":"dm\_alias

***Request - delete alias set:***

DELETE http://localhost:8080/acme-rest/repositories/REPO/alias-sets/6600000580000515 HTTP/1.1 Authorization: Basic ZG1hZG

***Response - delete alias set:***

HTTP/1.1 204 No Content

We are done here for this tutorial.

**Next:** [Documentum REST Extensibility Tutorial \(6\): Make Resource Query-able](#)

[Learn more about Documentum REST Services >>](#)