InCommon®

InCommon Certificate Manager

Custom Fields API

InCommon
c/o Internet2
1000 Oakbrook Drive, Suite 300
Ann Arbor MI, 48104

1 Introduction

InCommon CM allows you to add custom fields to device, SMIME and SSL certificates. This can be done via the InCommon CM interface or by API. This document explains how to add custom fields to certificates using the API.

- Prerequisite – Custom fields are enabled for your account.

2 Common Data Structures

CustomFieldDtoV2{

- id: int
- name: String
- certType: CertType
- state: CustomFieldState
- mandatory: List<CustomFieldArea>

}

CustomFieldArea{

- ADMIN_UI : string
- REST_API : string
- SOAP_API : string
- WEB_FORM : string

}

CustomFieldResponse {

- statusCode: int
- fields: List<CustomFieldDtoV2>

}

CertType {

- Device | SMIME | SSL

}

CustomFieldState {

- ACTIVE | INACTIVE

}

AuthData {

- login: String
- password: String
- customerLoginUri: String

}

Status Code – Possible Values
-100 = The URI of customer is invalid.
-102 = You are not authorized.
-130 = Custom fields are disabled for customer.
-131 = Custom fields limit exceeded for customer.
-132 = Custom field has no name.
-133 = Custom field has no certificate type.
-134 = Custom field has to have unique name.
-135 = Custom field isn't found.

3 Methods

3.1 Create

Create (

- authData: AuthData
- customField: CustomFieldDto

)

- *Name* and *certType* fields of *customField* parameter must not be empty. *Mandatory* and *state* fields can be empty, default value for them are *false* and *INACTIVE*. *Id* field of *customField* is ignored.
- Returns: CustomFieldResponse which contains newly created CustomFieldDto with *id* value or an error with StatusCode.

- Note: Combination of name and *certType* has to be unique for the customer.

Request:

```
<cus:createV2>
  <authData>
    <login>?</login>
    <password>?</password>
    <customerLoginUri>?</customerLoginUri>
  </authData>
  <customFieldDtoV2>
    <id>?</id>
    <name>?</name>
    <certType>?</certType>
     <!--Optional: zero or more repetitions-->
    <mandatory>?</mandatory>
    <state>?</state>
  </customFieldDtoV2>
</cus:createV2>
```

Response:

```
<ns2:createV2Response xmlns:ns2="http://customfield.ws.epki.comodo.com/">
  <return>
    <statusCode>0</statusCode>
    <customField>
      <id>?</id>
      <name>?</name>
     <!--Optional: zero or more repetitions-->
      <mandatory>?</mandatory>
      <certType>?</certType>
      <state>?</state>
    </customField>
  </return>
</ns2:createV2Response>
```

3.2 Read

Read (

- authData: AuthData
- certType: CertType

)

Returns: CustomFieldResponse with any number of CustomFieldDto from 0 to n (all existing for the customer). When *certType* parameter is specified, only custom fields of this type are returned. When *certType* parameter isn't specified, all existing custom fields of the customer are returned back.

Request:

```
<cus:readV2>
  <authData>
    <login>?</login>
    <password>?</password>
    <customerLoginUri>?</customerLoginUri>
  </authData>
  <!--Optional:-->
  <certType>?</certType>
</cus:readV2>
```

Response:

```
<ns2:readV2Response xmlns:ns2="http://customfield.ws.epki.comodo.com/">
  <return>
    <statusCode>0</statusCode>
    <customField>
      <id>?</id>
      <name>?</name>
     <!--Optional: zero or more repetitions-->
      <mandatory>?</mandatory>
      <certType>?</certType>
      <state>?</state>
    </customField>
  </return>
```

```
</ns2:readV2Response>
```

3.3 Update

Update (

- authData: AuthData
- customField: CustomFieldDtoV2

)

- Parameter *customField* must have *id* value. Other *customField* fields can have values or be empty. When the field value is empty, it won't be updated.
- Returns: CustomFieldResponse, which has updated CustomFieldDto or an error with StatusCode.
- Note: combination of *name* and *certType* has to be unique for the customer.

WARNING! Changing of *certType* of the custom field will delete all existing values of this field from customer's certificates (analog to existing behavior at customer's UI).

Request:

```
<cus:updateV2>
  <authData>
    <login>?</login>
    <password>?</password>
    <customerLoginUri>?</customerLoginUri>
  </authData>
  <customFieldDtoV2>
    <id>?</id>
    <!--Optional:-->
    <name>?</name>
    <certType>?</certType>
    <!--Optional: zero or more repetitions-->
    <mandatory>?</mandatory>
    <state>?</state>
  </customFieldDtoV2>
</cus:updateV2>
```

Response:

```
<ns2:updateV2Response xmlns:ns2="http://customfield.ws.epki.comodo.com/">
  <return>
    <statusCode>0</statusCode>
    <customField>
      <id>?</id>
      <name>?</name>
      <!--Optional: zero or more repetitions-->
      <mandatory>?</mandatory>
      <certType>?</certType>
      <state>?</state>
    </customField>
  </return>
</ns2:updateV2Response>
```

3.4 Delete

Delete (

- authData: AuthData
- customField: CustomFieldDto

)

- Parameter *customField* must have *id* value. Other fields values are ignored. The custom field with the specified *id* will be deleted or an error with StatusCode returned.

Request:

```
<cus:delete>
  <authData>
    <login>?</login>
    <password>?</password>
    <customerLoginUri>?</customerLoginUri>
  </authData>
```

```
      <customFieldDto>
        <id>76</id>
        <!--Ignored:-->
        <name>?</name>
        <mandatory>?</mandatory>
        <certType>?</certType>
        <state>?</state>
      </customFieldDto>
    </cus:delete>
```

Response:

```
    <ns2:deleteResponse xmlns:ns2="http://customfield.ws.epki.comodo.com/">
      <return>
        <statusCode>0</statusCode>
      </return>
    </ns2:deleteResponse>
```