This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

API Reference

Bulk Profile Import

12/14/2025

Bulk Profile Import

> Important
>
> **Prerequisites:** You need to enable profiles in UCS.

POST /profiles/import

Contents

Description

Imports several customer profiles in a single operation. Your application should use this operation if you need to import a large number of profiles into the UCS database.

- The operation supports the import of `.csv` files if the file is already uploaded on UCS' local file system.
- The operation manages:
  - Database consistency (data type conversion according to metadata).
  - Propagation of identification keys.
  - Backward compatibility with the APIs.

> **Warning**
> You cannot submit the profiles as a JSON array.

The operation's description of the supported format must be compliant with RFC4180:

- The first record must contain the column name for each field.
- There is one record per line and each line terminates with a line feed.
- Commas separate fields by default; if you wish to use another separator, set up the delimiter field of your operation.
- Spaces are considered as part of a field and will not be ignored (trimmed).

This operation returns the list of contacts which were not imported, with descriptions of the errors or ambiguities which prevented the server from adding them to the database.

Roles

If role-based access control is enabled in your environment, your application will need to be assigned the following roles in order to successfully perform a bulk import:

- UCS.Customer.executeBulkImport (Perform Bulk Import)
- UCS.Customer.createProfile (Create Customer Profile)
- UCS.Customer.createProfileExtension (Create Profile Extension)

Important

Your application must be assigned the `UCS.Customer.createProfileExtension` role only if profile extensions will be created by the bulk import.

Operation

Prerequisites:

- The Customer Profile Schema exists.

Your operation's header must comply with the following requirements:

- Content-Type is set to 'multipart/form-data'.
- Request is composed of a first part of type 'application/json', named 'description' and containing the JSON body as described below.

ID	CV.WS.PROF.IMPORT.1		
Method	POST		
URL	/profiles/import		
Parameter	**Type**	**Mandatory**	**Description**
Body			
format	"JSON" **Not supported** "CSV"	yes	Defines the format of the submitted data.
profiles	string	no	File path, for instance 'C:/profiles.csv'. Your application is responsible for uploading the file on the server's local filesystem.
id-key	string	no	A valid identification key. If specified, the server checks to see whether the profile already exists prior to the insertion. If the profile does exist in the database, the record containing the duplicate profile is skipped.
delimiter	char or "," by default	no	Defines the character used to separate the

ID	CV.WS.PROF.IMPORT.1		
			submitted profiles.
batch-size	integer 64 by default	no	The number of records to insert or update before a database COMMIT is issued (default to 64).
max-errors	integer	no	The number of errors to allow before ending the bulk load. An error can indicate that a given record was not parsed correctly, or that an ambiguity exists with an existing customer record (see response below).
encoding	string	no	Encoding character to use for file import. Default is UTF-8<ref> UTF-8 without BOM, see related bug http://bugs.sun.com/ view_bug.do;jsessionid=fdff2c415891f7
start-from-index	integer	no	The record number from which to start importing.
create-log	boolean	no	This flag turns extended import results logging on or off. The log data is written to an output file that has the same name as the input file, with an extension of `.log` appended. The default value is `false`.

Response

The Context Management Service API answers every request with an HTTP code. The following table shows the correct response for a successful request. See HTTP Response Codes and Errors for further details on the possible codes that this operation can return.

HTTP code	200
HTTP message	OK

Body	`{"errors": [`
	` {"index": '''$index_id1''', "reason":`
	`'''$reason_1'''},`
	` {"index": '''$index_id2''', "reason":`

HTTP code	200

	`'''$reason_2'''}],`
	`"ambiguous": [`
	` {"index": '''$index_id2''', "reason":`
	`'''$reason_3'''}]`
	`}`
	where:
	• **${index_idn}** is the index of the imported record in the source array.
	• "Ambiguous" lists the source data items which were ignored because of *Unresolved identity*: The given identification key is not uniquely identifying the customer; for instance, attempting to use "Name" to identify the customer, but there are more than one customer with the same name.
	• "Ambiguous" lists the source data items which were invalid for some reason. The reason description is one of the errors which can occur when inserting a profile (see Create Customer Profile).

Examples

Raw HTTP

Operation

```
POST /cv/profiles/import HTTP/1.1
Keep-Alive: 10
Content-Length: 320
Content-Type: multipart/form-data; boundary=zaHf8xb3LfVOXZ0o2KUyqWD4Zlfumh0R9uyMN
Host: localhost:9090
Connection: Keep-Alive
Expect: 100-Continue

--zaHf8xb3LfVOXZ0o2KUyqWD4Zlfumh0R9uyMN
Content-Disposition: form-data; name="description"
Content-Type: application/json; charset=UTF-8
Content-Transfer-Encoding: 8bit

{"delimiter":",","profiles":"D:/successContentProfiles.csv","format":"csv","max-errors":20}
--zaHf8xb3LfVOXZ0o2KUyqWD4Zlfumh0R9uyMN--
```

Here is a sample `.csv` file containing 10 records:

```
FirstName, LastName, PhoneNumber[2], PhoneNumber[1], EmailAddress, Address.AddressType,
Address.PostCode,Address.Address, Address.County, Address.City, Address.Country
 kertuleordi,gusch,1-355-530-4087,1-204-489-2084,kertuleordi@gusch.net,7,4703,"25,
et malesuada vulputate Proin",Pomona,Germany,Fullerton
 danye,hna,1-551-483-3810,1-455-658-9039,danye@hna.net,6,5101,"blabl \nDonec",Kailua,East
Timor,
Duluth
 almady,vanah,1-215-196-0375,1-632-596-1889,almady@vanah.net,0,4602,facilisis fermentum
lacus.,Irving,Cambodia,Jackson
 julaudricelsa,mosessaglowsket,1-363-017-4320,
1-600-591-2531,julaudricelsa@mosessaglowsket.net,
4,2644,id bibendum,Fort Lauderdale,Sudan,Orem
 clemara,gonebali,1-334-657-4832,1-226-703-2238,clemara@gonebali.net,6,6552,
aliquet rutrum nonummy pellentesque ac,Chula Vista,Jamaica,Waterbury
 lonni,mi,1-021-445-4631,1-716-116-8811,lonni@mi.net,7, 102,Duis vulputate lacus.
 felis eu eu et sagittis ut,Allentown,Russian Federation,North Charleston
 louignaelgarcecio,jecklettorneimermandetett,1-615-348-3144,1-161-040-
1742,louignaelgarcecio@jecklettorneimermandetett.net,1,6533,
nisl Suspendisse imperdiet,Jacksonville,Italy, Baltimore
 jantasie,jachils,1-324-661-2865,1-220-082-1681,jantasie@jachils.net,5,0171,neque lacus.
Donec mauris Donec facilisis metus. est,Columbia,
Central African Rep,Thousand Oaks
 hizellaighie,wollosch,1-207-086-4460,1-160-603-4275,hizellaighie@wollosch.net,
4,9653,Vestibulum ipsum eu sit,Albuquerque,Romania,Philadelphia
```

HTML Example

Operation

Consider the following code sample embedding the JSON request, which enables you to select the `.csv` file:

```
<html>
 <body>
 Insert JSON request in textbox below and select your CSV file.
 <form method="post" action="http://localhost:8080/cv/profiles/import">
 <br/>'application/json' import parameters:<br/>
 <TEXTAREA name="description" rows="10" cols="80">
{ profiles:"D:/myProfiles.csv", format:"csv", delimiter:",", batch-size:64, max-errors:20
}</TEXTAREA>
 <input type="submit" value="Send" />
 </form>
 </body>
 </html>
```

Result The service might return the following information:

```
{
  "errors": [
   {
    "index": 3,
    "reason": "Bad parameter 'Importing profile' reason : number of profile attributes
cannot be different than declared"
   },
   {
    "index": 4,
    "reason": "Bad parameter 'Invalid value' reason : \"no-integer\" is an invalid value for
type 'Integer'"
   }
  ]
  "ambiguous": [
   {
    "index": 12,
    "reason": "Some customer(s) are already matching this profile : idProfile1, idProfile2"
   },
   {
    "index": 14,
    "reason": "Some customer(s) are already matching this profile : idProfile15, idProfile232"
   }
  ]
}
```

Apache Example

The following code sample shows how to write an HTTP client for Apache which enables an import.

```
DefaultHttpClient client = new DefaultHttpClient();
HttpRequestRetryHandler retryHandler = new HttpRequestRetryHandler() {
 public boolean retryRequest(IOException exception, int executionCount, HttpContext context) {
 return false;
 }
};
client.setHttpRequestRetryHandler(retryHandler);



HttpPost httpPost = new HttpPost("http://localhost:9090/profiles/import");

JSONObject parameters = new JSONObject();
parameters.put("format", "csv");
parameters.put("profiles", "D:/myProfiles.csv");
parameters.put("delimiter", ",");
parameters.put("max-errors", 20);

MultipartEntity reqEntity = new MultipartEntity(HttpMultipartMode.STRICT);
reqEntity.addPart("description", new StringBody(parameters.toString(),
MediaType.APPLICATION_JSON.toString(), Charset.forName("UTF-8")));
httpPost.setEntity(reqEntity);

HttpResponse response = client.execute(httpPost);
HttpEntity resEntity = response.getEntity();

String res = EntityUtils.toString(resEntity, "UTF-8");

client.getConnectionManager().shutdown();
```

References

<references />