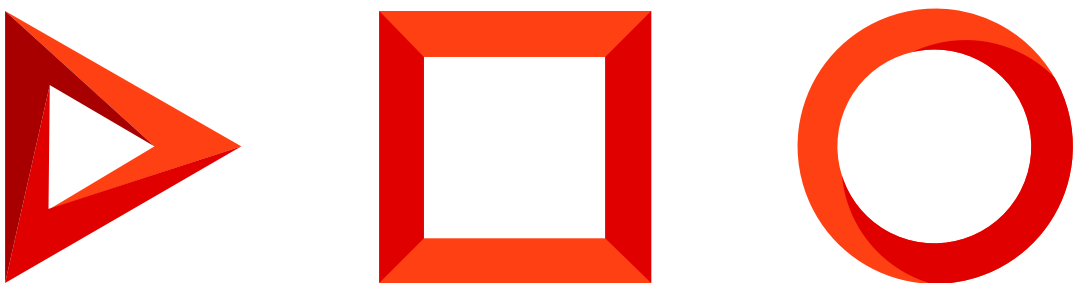
Creatio Academy

# Data services

## OData

Version 7.16



# Table of Contents

# OData

Advanced

**OData (Open Data Protocol)** is an ISO/IEC-approved OASIS standard. It defines a set of best practices for building and using REST API. Use OData to create REST-based services that let you publish and edit resources using simple HTTP requests. Such resources should be identified with a URL and defined in the data model.

The **purpose** of the OData protocol is to execute requests of external applications to the Creatio database server.

Creatio supports OData 4 and OData 3 protocols. OData 4 provides more features than OData 3. The main **difference** between the protocols is the data format of the server's response. Learn more about the differences between OData 3 and OData 4 protocols in the official OData documentation. Use OData 4 for Creatio integration.

All external requests to Creatio must be authenticated. We recommend using **Forms authentication** (cookie based) implemented using the AuthService.svc web service.

## OData 4 protocol

The `odata` web service provides access to Creatio objects via the OData 4 protocol.

odata service URL for .NET Framework

```
https://mycreatio.com/0/odata
```

odata service URL for .NET Core

```
https://mycreatio.com/odata
```

## OData 3 protocol

The `EntityDataService.svc` web service provides access to Creatio objects via the OData 3 protocol.

**`EntityDataService.svc` service URL**

```
https://mycreatio.com/0/ServiceModel/EntityDataService.svc
```

Use the `EntityDataService.svc` service to work with Creatio objects in a WCF client.

**Windows Communication Foundation (WCF)** is a program framework that handles data exchange between

applications. WCF is a part of .NET Framework.

The **WCF client operates** by receiving the service metadata and creating client proxy classes. The client application will communicate with Creatio's `EntityDataService.svc` service using these mediator classes.

To **implement the client application**:

1. Create a .NET project where you will implement the Creatio integration.
2. Generate client proxy classes for the `EntityDataService.svc` service.
3. Create a context instance of the `EntityDataService.svc` service's runtime environment.
4. Implement the integration's client business logic using the methods of the proxy class instance.

There are several **ways to generate** the proxy classes for the `EntityDataService.svc` service:

- Using the `DataServiceModel Metadata Utility Tool (DataSvcutil.exe)`.
- From the Visual Studio client application project.

## Generate proxy classes using the `DataServiceModel Metadata Utility Tool`

`DataSvcUtil.exe` is a command line program provided by `WCF Data Services`. The utility uses the OData channel and generates the data service's client classes required to access the service from the .NET Framework client application. The data classes are generated using the following **metadata sources**:

- `WSDL` is the service metadata document. Describes the data model the data service provides.
- `CSDL` is the data model file. Uses the common schema definition language (`CSDL`). Learn more: [MC-CSDL]: Conceptual Schema Definition File Format.
- `EDMX` is an *.xml file. Create it using programs for working with the `EDM` model. The programs are included in the `Entity Framework`. Learn more: [MC-EDMX]: Entity Data Model for Data Services Packaging Format.

The `DataSvcUtil.exe` tool is usually installed in the `C:\Windows\Microsoft.NET\Framework\v4.0` directory. For 64-bit systems, the respective directory is usually `C:\Windows\Microsoft.NET\Framework64\v4.0`.

**The `DataSvcutil.exe` utility calling format**

```
datasvcutil /out:file [/in:file | /uri:serviceuri] [/dataservicecollection] [/language:devlang]
```

Find the detailed information about the `DataSvcutil.exe` utility in the official MSDN documentation.

## Generate proxy classes from the Visual Studio client application project

To generate proxy classes from **the Visual Studio client application project**:

1. Right-click the project where you want to implement Creatio integration and select [ *Add Service Reference...* ] in the context menu.
2. Enter the complete address of the `EntityDataService.svc` service in the [ *Address* ] field.

3. Click [ *Go* ] and specify the Creatio user credentials. The entities supported by the service will appear in the [ *Services* ] window upon success.
4. Specify the namespace to include the generated proxy classes in the [ *Namespace* ] field. For example, `CreatioServiceReference`. You will also need to link this namespace in the `using` block of the project.
5. Click [ *OK* ] to generate the proxy classes. This will add a new `Reference.cs` code file with the description of proxy classes to the project. Use the classes to call and interact with the data service's resources as objects.

Visual Studio also lets you generate the service's proxy classes from the **service metadata file** stored on your drive. To do this, enter the complete path to the metadata file, starting from the `file://` prefix, in the [ *Address* ] field on step 2.

```
file://C:/metadata.xml
```

Visual Studio will link the `Microsoft.Data.Services.Client.dll` build in the project upon generating the service's proxy classes. This ensures OData 3 protocol support. If the client application requires an earlier version of the protocol, link the corresponding build manually. This client library lets you call the `EntityDataService.svc` data service using the standard .NET Framework programming templates and the LINQ request language.

**Add** the `using` directives to and declare the variable of the OData service URL in the project code to ensure successful compilation.

**`using` directives**

```
using System;
using System.Data.Services.Client;
using System.Net;
using Terrasoft.Sdk.Examples.CreatioServiceReference;
using System.Linq;
```

**Declare the variable of the OData service URL**

```
private static Uri serverUri = new Uri("http://<server_name>/<application_name>/0/ServiceModel/E
```

## Limitations of OData protocol

When using the OData protocol, keep in mind the following **limitations**:

- It is not possible to create system users.
- It is not possible to specify the culture of the returned data. The culture is determined by the culture of the user on whose behalf you executed the request.
- The response body can contain up to 20 000 lines.
- A batch request can contain up to 100 sub-requests.

- The [ *MaxFileSize* ] system setting controls the maximum size of the files you can upload using requests. The default value is 10 Mb.

# OData integration examples

Advanced

View the Creatio API documentation that has examples of various CRUD operations via OData 3 and OData 4 protocols on the Postman website.

# Stream data type request examples

Advanced

The following **elements** have the Stream data type:

- Images
- Files
- Binaries

Use the following standard **methods** to work with the Stream data type:

- `GET` – retrieve data
- `POST` – add data
- `PUT` – modify data
- `DELETE` – delete data

Clear the browser cache to display Creatio request results when working with the Stream data type.

## Retrieve data

**Example.** Use the OData service to retrieve the “New user” contact's photo.

### Implement the example

1. Retrieve the identifier of the “New user” contact's photo.

   The contact's photo is stored in the `[Data]` column of the `[SysImage]` database table. Run the following SQL query to **retrieve the identifier of the “New user” contact's photo**.

   SQL query

   ```
   select Id from SysImage where Id = (select PhotoId from Contact where Name = 'New user')
   ```

Response

```
29FE7EDF-4DB9-4E09-92B0-018047BA1F71
```

2. Retrieve the “New user” contact's photo.

   Execute the following request to **retrieve the “New user” contact's photo**.

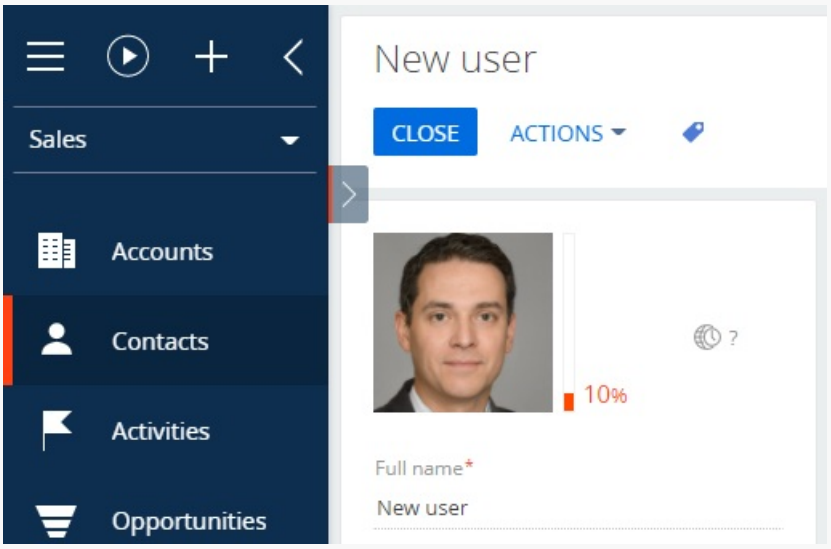
Request

```
// Retrieve the [Data] field value for the 29FE7EDF-4DB9-4E09-92B0-018047BA1F71 [Id] object i
GET http://mycreatio.com/0/odata/SysImage(29FE7EDF-4DB9-4E09-92B0-018047BA1F71)/Data
```

Response

```
Status: 200 OK
```

Results in Creatio



# Add data

**Example.** Use the OData service to add the “New user” contact. Then, add a photo to the contact.

## Implement the example

1. Add the “New user” contact.

   Creatio stores all contacts in the `[Contact]` database table. Execute the following request to **add the “New user” contact**.

   Request

   ```
   // Add an object instance to the [Contact] collection.
   POST http://mycreatio.com/0/odata/Contact

   Accept: application/json; odata=verbose
   Content-Type: application/json; odata=verbose; IEEE754Compatible=true
   BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO

   {
       // Write the "New user" contact name to the [Name] field.
       "Name": "New user"
   }
   ```

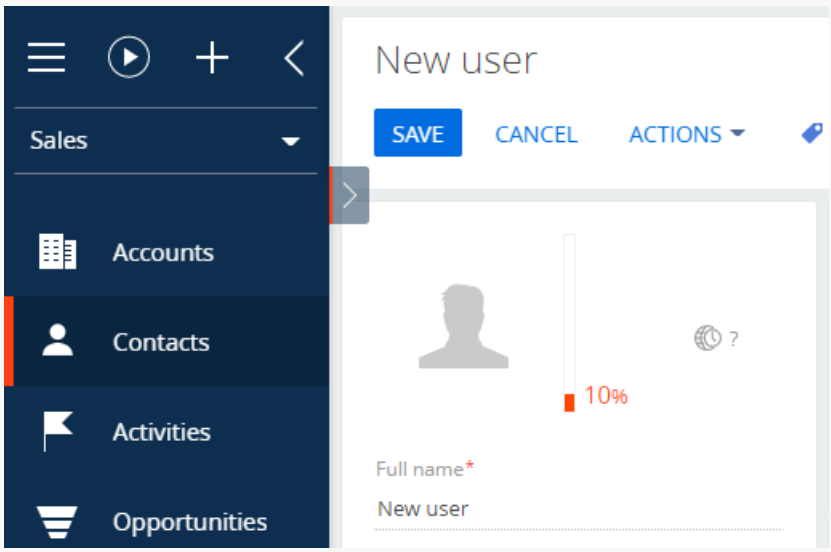
   Response

   ```
   Status: 201 Created

   {
       "@odata.context": "http://mycreatio.com/0/odata/$metadata#Contact/$entity",
       "Id": "4c63c8fa-467b-48a6-973f-b2069298404f",
       "Name": "New user",
       "OwnerId": "410006e1-ca4e-4502-a9ec-e54d922d2c00",
       "CreatedOn": "2021-01-14T08:33:29.009023Z",
   ```

```
    "CreatedById": "410006e1-ca4e-4502-a9ec-e54d922d2c00",
    "ModifiedOn": "2021-01-14T08:33:29.009023Z",
    "ModifiedById": "410006e1-ca4e-4502-a9ec-e54d922d2c00",
    "ProcessListeners": 0,
    "Dear": "",
    "SalutationTypeId": "00000000-0000-0000-0000-000000000000",
    "GenderId": "00000000-0000-0000-0000-000000000000",
    "AccountId": "00000000-0000-0000-0000-000000000000",
    "DecisionRoleId": "00000000-0000-0000-0000-000000000000",
    "TypeId": "00000000-0000-0000-0000-000000000000",
    "JobId": "00000000-0000-0000-0000-000000000000",
    "JobTitle": "",
    "DepartmentId": "00000000-0000-0000-0000-000000000000",
    "BirthDate": "0001-01-01T00:00:00Z",
    "Phone": "",
    "MobilePhone": "",
    "HomePhone": "",
    "Skype": "",
    "Email": "",
    "AddressTypeId": "00000000-0000-0000-0000-000000000000",
    "Address": "",
    "CityId": "00000000-0000-0000-0000-000000000000",
    "RegionId": "00000000-0000-0000-0000-000000000000",
    "Zip": "",
    "CountryId": "00000000-0000-0000-0000-000000000000",
    "DoNotUseEmail": false,
    "DoNotUseCall": false,
    "DoNotUseFax": false,
    "DoNotUseSms": false,
    "DoNotUseMail": false,
    "Notes": "",
    "Facebook": "",
    "LinkedIn": "",
    "Twitter": "",
    "FacebookId": "",
    "LinkedInId": "",
    "TwitterId": "",
    "ContactPhoto@odata.mediaEditLink": "Contact(4c63c8fa-467b-48a6-973f-b2069298404f)/Contac
    "ContactPhoto@odata.mediaReadLink": "Contact(4c63c8fa-467b-48a6-973f-b2069298404f)/Contac
    "ContactPhoto@odata.mediaContentType": "application/octet-stream",
    "TwitterAFDAId": "00000000-0000-0000-0000-000000000000",
    "FacebookAFDAId": "00000000-0000-0000-0000-000000000000",
    "LinkedInAFDAId": "00000000-0000-0000-0000-000000000000",
    "PhotoId": "00000000-0000-0000-0000-000000000000",
    "GPSN": "",
    "GPSE": "",
    "Surname": "user",
    "GivenName": "New",
    "MiddleName": "",
```

```
    "Confirmed": true,
    "IsNonActualEmail": false,
    "Completeness": 0,
    "LanguageId": "6ebc31fa-ee6c-48e9-81bf-8003ac03b019",
    "Age": 0
}
```

Results in Creatio




The “New user” contact identifier is “4c63c8fa-467b-48a6-973f-b2069298404f.”

2. Add a photo to the “New user” contact.

   The contact photo must be stored in the `[Data]` column of the `[SysImage]` database table. Since there is no table record for this contact, you must add it. Execute the following request to **add a record to the table**.

Request

```
// Add an object instance to the [SysImage] collection.
POST http://mycreatio.com/0/odata/SysImage

Accept: application/json; odata=verbose
Content-Type: application/json; odata=verbose; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO

{
    // Write the filename of the contact photo to the [Name] field.
    "Name": "scr_NewContactPhoto.png",
    // Write an arbitrary record identifier to the [Id] field of the [SysImage] table.
    "Id": "410006E1-CA4E-4502-A9EC-E54D922D2C01",
    // Write the file type of the contact photo to the [MimeType] field.
```

```
    "MimeType": "image/png"
}
```

Response

```
Status: 201 Created

{
    "@odata.context": "http://mycreatio.com/0/odata/$metadata#SysImage/$entity",
    "Id": "410006e1-ca4e-4502-a9ec-e54d922d2c01",
    "CreatedOn": "2021-01-14T08:52:47.7573789Z",
    "CreatedById": "410006e1-ca4e-4502-a9ec-e54d922d2c00",
    "ModifiedOn": "2021-01-14T08:52:47.7573789Z",
    "ModifiedById": "410006e1-ca4e-4502-a9ec-e54d922d2c00",
    "ProcessListeners": 0,
    "UploadedOn": "0001-01-01T00:00:00Z",
    "Name": "scr_NewContactPhoto.png",
    "Data@odata.mediaEditLink": "SysImage(410006e1-ca4e-4502-a9ec-e54d922d2c01)/Data",
    "Data@odata.mediaReadLink": "SysImage(410006e1-ca4e-4502-a9ec-e54d922d2c01)/Data",
    "Data@odata.mediaContentType": "application/octet-stream",
    "MimeType": "image/png",
    "HasRef": false,
    "PreviewData@odata.mediaEditLink": "SysImage(410006e1-ca4e-4502-a9ec-e54d922d2c01)/Previe
    "PreviewData@odata.mediaReadLink": "SysImage(410006e1-ca4e-4502-a9ec-e54d922d2c01)/Previe
    "PreviewData@odata.mediaContentType": "application/octet-stream"
}
```

The record was added to the `[SysImage]` database table, however the value of the `[Data]` column is “0x.”

Pass the image in the request body. The image filename must match the value of the `[Name]` field. Execute the following request to **add the contact photo** to the `[Data]` column.

Request

```
// Update the value of the [Data] field for the 410006e1-ca4e-4502-a9ec-e54d922d2c01 [Id] obj
PUT http://mycreatio.com/0/odata/SysImage(410006e1-ca4e-4502-a9ec-e54d922d2c01)/Data

Accept: application/json; text/plain; */*
Content-Type: application/octet-stream; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
```

Response

```
Status: 200 OK
```

3. Bind the added photo to the “New user” contact.

   Link the `[Data]` field of the `[SysImage]` table to the `[PhotoId]` field of the `[Contact]` table to bind the photo to the “New user” contact. Execute the following request to **set up the binding**.

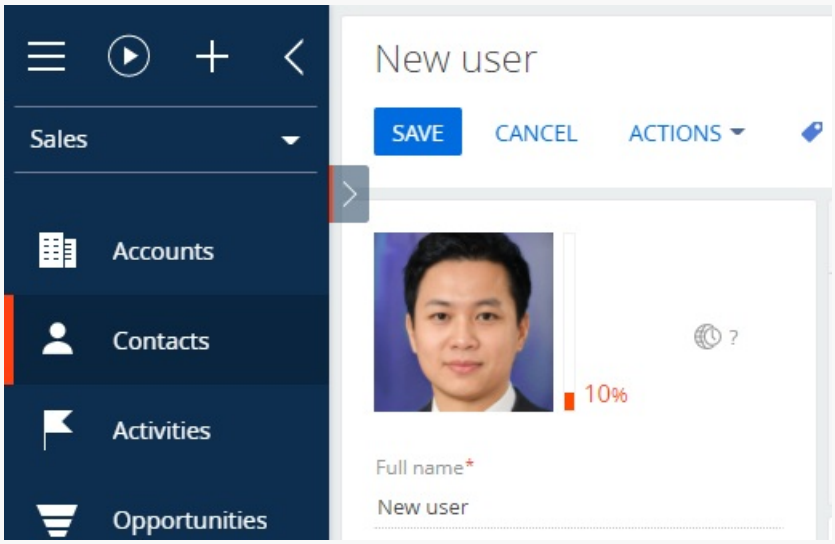
Request

```
// Update the value of the [PhotoId] field for the 4c63c8fa-467b-48a6-973f-b2069298404f [Id]
PATCH http://mycreatio.com/0/odata/Contact(4c63c8fa-467b-48a6-973f-b2069298404f)

Accept: application/json;odata=verbose
Content-Type: application/json; odata=verbose; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO

{
    // Write the identifier of the record in the [SysImage] table to the [PhotoId] field.
    "PhotoId": "410006e1-ca4e-4502-a9ec-e54d922d2c01"
}
```

Response

```
Status: 204 No Content
```

Results in Creatio




Execute the following to **add a photo to an existing contact**:

1. A `POST` request to add an object instance to the `[SysImage]` collection.
2. A `PUT` request to update the value of the `[Data]` field for the object instance in the `[SysImage]` collection.
3. A `PATCH` request to bind the added photo to the “New user” contact.

# Modify data

**Example.** Use the OData service to update the photo of the existing “New user” contact.

## Implement the example

1. Retrieve the identifier of the “New user” contact's photo.

   The contact's photo is stored in the `[Data]` column of the `[SysImage]` database table. Run the following SQL query to **retrieve the identifier of the “New user” contact's photo**.

   SQL query

   ```
   select Id from SysImage where Id = (select PhotoId from Contact where Name = 'New user')
   ```

Response

```
29FE7EDF-4DB9-4E09-92B0-018047BA1F71
```

2. Update the "`New user`" contact's photo.

   Execute the following request to **update the "New user" contact's photo**.

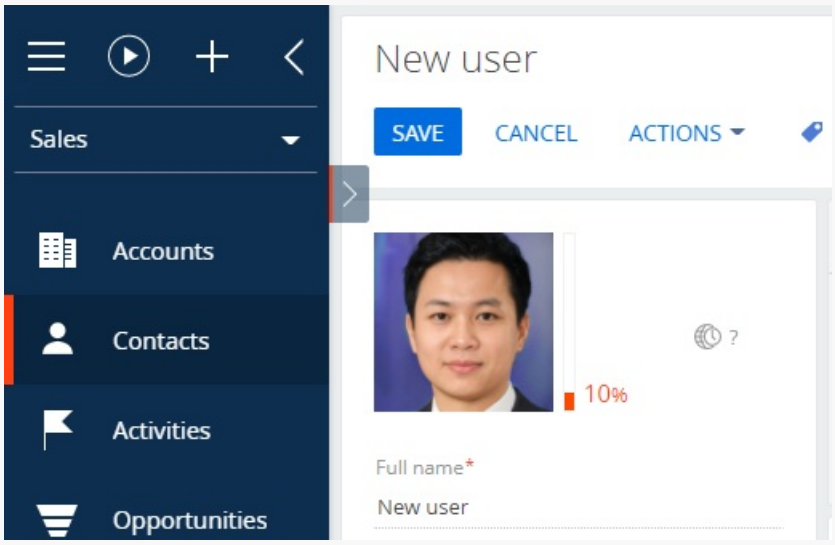
Request

```
// Update the [Data] field for the 29FE7EDF-4DB9-4E09-92B0-018047BA1F71 [Id] object instance
PUT http://mycreatio.com/0/odata/SysImage(29FE7EDF-4DB9-4E09-92B0-018047BA1F71)/Data

Accept: application/json; text/plain; */*
Content-Type: application/octet-stream; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
```

Response

```
Status: 200 OK
```

Results in Creatio




# Delete data

**Example.** Use the OData service to delete the photo of the “New user” contact.

## Implement the example

1. Retrieve the identifier of the “New user” contact's photo.

   The contact's photo is stored in the `[Data]` column of the `[SysImage]` database table. Run the following SQL query to **retrieve the identifier of the “New user” contact's photo**.

   SQL query

   ```
   select Id from SysImage where Id = (select PhotoId from Contact where Name = 'New user')
   ```

   Response

   ```
   29FE7EDF-4DB9-4E09-92B0-018047BA1F71
   ```

2. Delete the “New user” contact's photo.

   Execute the following request to **delete the “New user” contact's photo**.

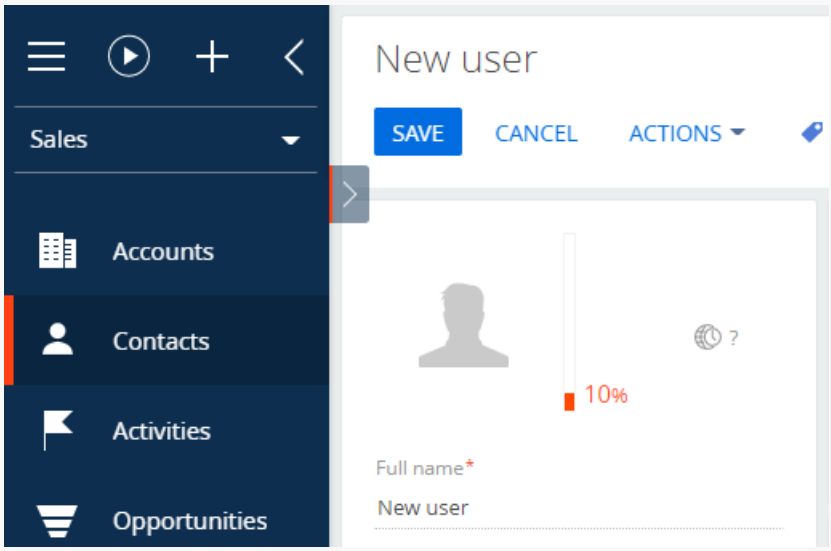
   Request

```
// Delete the value of the [Data] field for the 29FE7EDF-4DB9-4E09-92B0-018047BA1F71 [Id] obj
DELETE http://mycreatio.com/0/odata/SysImage(29FE7EDF-4DB9-4E09-92B0-018047BA1F71)/Data

Accept: application/json; text/plain; */*
Content-Type: application/json; charset=utf-8; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
```

Response

```
Status: 204 No Content
```

Results in Creatio




# Batch request examples

Advanced

**Batch requests** combine multiple HTTP requests by specifying each request as a separate object in the batch request's body. The Creatio database server returns a single HTTP response that contains the responses to each request. Use batch requests to improve Creatio performance.

The batch requests utilize:

- The `POST` HTTP method.
- The $batch parameter.
- The `Content-Type` header.

The **values** of the `Content-Type` header:

- `application/json` – restricts the content the server returns for each request within the batch request to a single type.
- `multipart/mixed` – allows you to set unique `Content-Type` headers for each request in the batch request body.

If one of the requests completes with a 4xx-5xx group response code, the subsequent requests will not be executed. Add the Prefer: `continue-on-error` header to the main HTTP request to enable the execution of the subsequent requests.

**Attention.** A batch request can contain up to 100 requests.

## Batch request (`Content-Type: application/json`)

Batch request

```
POST http://mycreatio.com/0/odata/$batch

Content-Type: application/json; odata=verbose; IEEE754Compatible=true
Accept: application/json
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
ForceUseSession: true

{
    "requests": [
    {
        // Add an object instance to the City collection.
        "method": "POST",
        "url": "City",
        "id": "t3",
        "body": {
            // Add the Burbank value to the Name field.
            "Name": "Burbank"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
        }
    },
    {
        // Add an object instance to the City collection.
        "method": "POST",
        "atomicityGroup": "g1",
        "url": "City",
```

```
        "id": "t3",
        "body": {
            // Add the 62f9bc01-57cf-4cc7-90bf-8672acc922e3 value to the Id field.
            "Id": "62f9bc01-57cf-4cc7-90bf-8672acc922e3",
            // Add the Spokane value to the Name field.
            "Name": "Spokane"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
        }
    },
    {
        // Update the 62f9bc01-57cf-4cc7-90bf-8672acc922e3 id object instance in the City collec
        "method": "PATCH",
        "atomicityGroup": "g1",
        "url": "City/62f9bc01-57cf-4cc7-90bf-8672acc922e3",
        "id": "t2",
        "body": {
            // Change the value of the Name field to Texas.
            "Name": "Texas"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
        }
    }
    ]
}
```

Response

Status: 200 OK

```
{
    "responses": [
        {
            "id": "t3",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(b6a05348-55b1-4314-a228-436ba305
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
```

```
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "b6a05348-55b1-4314-a228-436ba305d2f3",
                "CreatedOn": "2020-05-18T17:50:39.2838673Z",
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T17:50:39.2838673Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Burbank",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        },
        {
            "id": "t3",
            "atomicityGroup": "c59e36b2-2aa9-44fa-86d3-e7d68eecbfa0",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(62f9bc01-57cf-4cc7-90bf-8672acc9
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "62f9bc01-57cf-4cc7-90bf-8672acc922e3",
                "CreatedOn": "2020-05-18T17:50:39.361994Z",
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T17:50:39.361994Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Spokane",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        },
        {
            "id": "t2",
            "atomicityGroup": "c59e36b2-2aa9-44fa-86d3-e7d68eecbfa0",
            "status": 204,
            "headers": {
                "odata-version": "4.0"
            }
        }
    ]
}
```

## Batch request (`Content-Type: application/json` and `Prefer: continue-on-error`)

Batch request

```
POST http://mycreatio.com/0/odata/$batch

Content-Type: application/json; odata=verbose; IEEE754Compatible=true
Accept: application/json
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
ForceUseSession: true
Prefer: continue-on-error

{
    "requests": [
    {
        // Add an object instance to the City collection.
        "method": "POST",
        "url": "City",
        "id": "t3",
        "body": {
            // Add the Burbank value to the Name field.
            "Name": "Burbank"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
        }
    },
    {
        // Update the 62f9bc01-57cf-4cc7-90bf-8672acc922e3 id object instance in the City collec
        "method": "PATCH",
        "atomicityGroup": "g1",
        "url": "City/62f9bc01-57cf-4cc7-90bf-8672acc922e2",
        "id": "t2",
        "body": {
            // Change the value of the Name field to Indiana.
            "Name": "Indiana"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
```

```
        }
    },
    {
        // Add an object instance to the City collection.
        "method": "POST",
        "atomicityGroup": "g1",
        "url": "City",
        "id": "t3",
        "body": {
            // Write the 62f9bc01-57cf-4cc7-90bf-8672acc922a1 value to the Id field.
            "Id": "62f9bc01-57cf-4cc7-90bf-8672acc922a1",
            // Write the Iowa value to the Name field.
            "Name": "Iowa"
        },
        "headers": {
            "Content-Type": "application/json;odata=verbose",
            "Accept": "application/json;odata=verbose",
            "Prefer": "continue-on-error"
        }
    }
    ]
}
```

Response

Status: 200 OK

```
{
    "responses": [
        {
            "id": "t3",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(2f5e68f8-38bd-43c1-8e15-a2f13b0a
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "2f5e68f8-38bd-43c1-8e15-a2f13b0aa56a",
                "CreatedOn": "2020-05-18T18:06:50.7329808Z",
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T18:06:50.7329808Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Burbank",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
```

```
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        },
        {
            "id": "t2",
            "atomicityGroup": "0c1c4019-b9fb-4fb3-8642-2d0660c4551a",
            "status": 204,
            "headers": {
                "odata-version": "4.0"
            }
        },
        {
            "id": "t3",
            "atomicityGroup": "0c1c4019-b9fb-4fb3-8642-2d0660c4551a",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(62f9bc01-57cf-4cc7-90bf-8672acc9
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "62f9bc01-57cf-4cc7-90bf-8672acc922a1",
                "CreatedOn": "2020-05-18T18:06:50.7954775Z",
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T18:06:50.7954775Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Iowa",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        }
    ]
}
```


## Batch request (`Content-Type: multipart/mixed`)

Batch request

```
POST http://mycreatio.com/0/odata/$batch
```

```
Content-Type: multipart/mixed;boundary=batch_a685-9724-d873; IEEE754Compatible=true
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
ForceUseSession: true

--batch_a685-9724-d873
Content-Type: multipart/mixed; boundary=changeset_06da-d998-8e7e

--changeset_06da-d998-8e7e
Content-Type: application/http
Content-Transfer-Encoding: binary

// Add an object instance to the City collection.
POST City HTTP/1.1
Content-ID: 1
Accept: application/atomsvc+xml;q=0.8, application/json;odata=verbose;q=0.5, */*;q=0.1
Content-Type: application/json;odata=verbose

// Write the Gilbert value to the Name field.
{"Name": "Gilbert"}

--changeset_06da-d998-8e7e
Content-Type: application/http
Content-Transfer-Encoding: binary

// Update the 62f9bc01-57cf-4cc7-90bf-8672acc922e2 id object instance in the City collection.
PATCH City/62f9bc01-57cf-4cc7-90bf-8672acc922e2 HTTP/1.1
Content-ID: 2
Accept: application/atomsvc+xml;q=0.8, application/json;odata=verbose;q=0.5, */*;q=0.1
Content-Type: application/json;odata=verbose

// Change the value of the Name field to Lincoln.
{"Name": "Lincoln"}

--changeset_06da-d998-8e7e
Content-Type: application/http
Content-Transfer-Encoding: binary

// Delete the 62f9bc01-57cf-4cc7-90bf-8672acc922e2 id object instance from the City collection.
DELETE City/62f9bc01-57cf-4cc7-90bf-8672acc922e2 HTTP/1.1
Content-ID: 3
Accept: application/atomsvc+xml;q=0.8, application/json;odata=verbose;q=0.5, */*;q=0.1
Content-Type: application/json;odata=verbose
```

Response

```
Status: 200 OK

--batchresponse_e17aace9-5cbb-49bd-b7ad-f1be8cc8c9d8
Content-Type: multipart/mixed; boundary=changesetresponse_a08c1df6-4b82-4a9b-be61-7ef4cc7b23ba

--changesetresponse_a08c1df6-4b82-4a9b-be61-7ef4cc7b23ba
Content-Type: application/http
Content-Transfer-Encoding: binary
Content-ID: 1

HTTP/1.1 201 Created
Location: https://mycreatio.com/0/odata/City(fbd0565f-fa8a-4214-ae89-c976c5f3acb4)
Content-Type: application/json; odata.metadata=minimal
OData-Version: 4.0

{
    "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
    "Id": "fbd0565f-fa8a-4214-ae89-c976c5f3acb4",
    "CreatedOn": "2020-05-18T18:41:57.0917235Z",
    "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
    "ModifiedOn": "2020-05-18T18:41:57.0917235Z",
    "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
    "Name": "Gilbert",
    "Description": "",
    "CountryId": "00000000-0000-0000-0000-000000000000",
    "RegionId": "00000000-0000-0000-0000-000000000000",
    "TimeZoneId": "00000000-0000-0000-0000-000000000000",
    "ProcessListeners": 0
}
--changesetresponse_a08c1df6-4b82-4a9b-be61-7ef4cc7b23ba
Content-Type: application/http
Content-Transfer-Encoding: binary
Content-ID: 2

HTTP/1.1 204 No Content
OData-Version: 4.0


--changesetresponse_a08c1df6-4b82-4a9b-be61-7ef4cc7b23ba
Content-Type: application/http
Content-Transfer-Encoding: binary
Content-ID: 3

HTTP/1.1 204 No Content


--changesetresponse_a08c1df6-4b82-4a9b-be61-7ef4cc7b23ba--
```

```
--batchresponse_e17aace9-5cbb-49bd-b7ad-f1be8cc8c9d8--
```

## Batch request (`Content-Type: multipart/mixed` and different sets of requests)

Batch request

```
POST http://mycreatio.com/0/odata/$batch

Content-Type: multipart/mixed;boundary=batch_a685-9724-d873; IEEE754Compatible=true
Accept: application/json
BPMCSRF: OpK/NuJJ1w/SQxmPvwNvfO
ForceUseSession: true

--batch_a685-9724-d873
Content-Type: multipart/mixed; boundary=changeset_06da-d998-8e7e

--changeset_06da-d998-8e7e
Content-Type: application/http
Content-Transfer-Encoding: binary

// Add an object instance to the City collection.
POST City HTTP/1.1
Content-ID: 1
Accept: application/atomsvc+xml;q=0.8, application/json;odata=verbose;q=0.5, */*;q=0.1
Content-Type: application/json;odata=verbose

// Write the d6bc67b1-9943-4e47-9aaf-91bf83e9c285 value to the Id field.
// Write the Nebraska value to the Name field.
{"Id": "d6bc67b1-9943-4e47-9aaf-91bf83e9c285", "Name": "Nebraska"}

--batch_a685-9724-d873
Content-Type: multipart/mixed; boundary=changeset_06da-d998-8e71

--changeset_06da-d998-8e71
Content-Type: application/http
Content-Transfer-Encoding: binary

// Add an object instance to the City collection.
POST City HTTP/1.1
Content-ID: 2
Accept: application/atomsvc+xml;q=0.8, application/json;odata=verbose;q=0.5, */*;q=0.1
Content-Type: application/json;odata=verbose
```

```
// Write the d6bc67b1-9943-4e47-9aaf-91bf83e9c286 value to the Id field.
// Add the Durham value to the Name field.
{"Id": "d6bc67b1-9943-4e47-9aaf-91bf83e9c286", "Name": "Durham"}
```

Response

Status: 200 OK


```
{
    "responses": [
        {
            "id": "1",
            "atomicityGroup": "e9621f72-42bd-47c1-b271-1027e4b68e3b",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(d6bc67b1-9943-4e47-9aaf-91bf83e9
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "d6bc67b1-9943-4e47-9aaf-91bf83e9c285",
                "CreatedOn": "2020-05-18T18:49:16.3766324Z",
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T18:49:16.3766324Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Nebraska",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        },
        {
            "id": "2",
            "atomicityGroup": "960e2272-d8cb-4b4d-827c-0181485dd71d",
            "status": 201,
            "headers": {
                "location": "https://mycreatio.com/0/odata/City(d6bc67b1-9943-4e47-9aaf-91bf83e9
                "content-type": "application/json; odata.metadata=minimal",
                "odata-version": "4.0"
            },
            "body": {
                "@odata.context": "https://mycreatio.com/0/odata/$metadata#City/$entity",
                "Id": "d6bc67b1-9943-4e47-9aaf-91bf83e9c286",
                "CreatedOn": "2020-05-18T18:49:16.4078852Z",
```

```
                "CreatedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "ModifiedOn": "2020-05-18T18:49:16.4078852Z",
                "ModifiedById": "dad159f3-6c2d-446a-98d2-0f4d26662bbe",
                "Name": "Durham",
                "Description": "",
                "CountryId": "00000000-0000-0000-0000-000000000000",
                "RegionId": "00000000-0000-0000-0000-000000000000",
                "TimeZoneId": "00000000-0000-0000-0000-000000000000",
                "ProcessListeners": 0
            }
        }
    ]
}
```

# WCF client request examples

Advanced

Use the DataServiceQuery universal class to retrieve the service's object collection. This class is a request to the service that retrieves the collection of a specific type of entities.

Create a context object instance of the Creatio application environment to execute a request to `EntityDataService.svc`.

The examples in this article will use the forms authentication.

To **implement the forms authentication**:

1. Create a `LoginClass` class.
2. Implement the `authServiceUri` (a string that requests the `Login` method of the `AuthService.svc` authentication service) and `AuthCookie` (Creatio's authentication cookies) fields.
3. Implement the `TryLogin(string userName, string userPassword)` method that authenticates the user and saves the server's response to the `AuthCookie` field.
4. Implement the `OnSendingRequestCookie(object sender, SendingRequestEventArgs e)` method that will be called in response to an event of the `SendingRequest` context instance (creating a new `HttpWebRequest` instance).

   The `OnSendingRequestCookie` method authenticates the user and adds the cookies received in response to the data request.

   **`OnSendingRequestCookie`**

```
static void OnSendingRequestCookie(object sender, SendingRequestEventArgs e)
{
    // Call the method of the LoginClass class that authenticates the user's method passed in
    LoginClass.TryLogin("CreatioUserName", "CreatioUserPassword");
    var req = e.Request as HttpWebRequest;
```

```
        // Add the received authentication cookies to the data request.
        req.CookieContainer = LoginClass.AuthCookie;
        e.Request = req;
    }
```

There are several **ways to execute** the service request:

- A LINQ request to the named `DataServiceQuery` object received from the service context.
- Implicit enumeration of the `DataServiceQuery` object received from the service context.
- Explicit call of the Execute method of the `DataServiceQuery` object. Call the BeginExecute method for asynchronous execution.

# Retrieve a contact collection via a LINQ request.

**Example.** Define and implement the LINQ request that returns all contact entities of the `EntityDataService.svc` service.

## Implement the example

**The LINQ request**

```
public static void GetOdataCollectioByLinqWcfExample()
{
    // Create the Creatio application context.
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    try
    {
        // Build a LINQ request to retrieve the contact collection.
        var allContacts = from contacts in context.ContactCollection
            select contacts;
        foreach (Contact contact in allContacts)
        {
            // Execute the contact actions.
        }
    }
    catch (Exception ex)
    {
        // Process errors.
    }
}
```

# Retrieve a contact collection via an implicit request

**Example.** Use context to execute an implicit request that retrieves all contact entities of the `EntityDataService.svc` service.

## Implement the example

**`GetOdataCollectionByImplicitRequestExample()`**

```
public static void GetOdataCollectionByImplicitRequestExample()
{
    // Create the Creatio application context.
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    try
    {
        // Define an implicit request that retrieves the contact collection from the service.
        DataServiceQuery<Contact> allContacts = context.ContactCollection;
        foreach (Contact contact in allContacts)
        {
            // Execute the actions with contacts.
        }
    }
    catch (Exception ex)
    {
        // Process errors.
    }
}
```

# Retrieve a contact collection via an explicit request

**Example.** Use the DataServiceContext context to execute an explicit request that retrieves all contact entities of the `EntityDataService.svc` service.

## Implement the example

**`GetOdataCollectionByExplicitRequestExample()`**

```
public static void GetOdataCollectionByExplicitRequestExample()
```

```
{
    // Define the Uri of the service request that returns the contact collection.
    Uri contactUri = new Uri(serverUri, "ContactCollection");
    // Create the Creatio application context
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    try
    {
        // Call the Execute<>() method to execute an explicit request to the service.
        foreach (Contact contact in context.Execute<Contact>(contactUri))
        {
            // Execute the contact actions.
        }
    }
    catch (Exception ex)
    {
        // Process errors.
    }
}
```

## CRUD operation examples

Retrieve an object

```
public static void GetOdataObjectByWcfExample()
{
    // Create the Creatio application context.
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    //
    var contact = context.ContactCollection.Where(c => c.Name.Contains("User")).First();
    // Execute the contact actions.
}
```

Create an object

```
public static void CreateCreatioEntityByOdataWcfExample()
{
    // Create a new contact and initialize its properties.
    var contact = new Contact()
    {
        Id = Guid.NewGuid(),
```

```
        Name = "New Test User"
    };
    // Create a new account to which the contact is connected and initialize the account's prope
    var account = new Account()
    {
        Id = Guid.NewGuid(),
        Name = "Some Company"
    };
    contact.Account = account;
    // Create the Creatio application context.
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>1(OnSendingRequestCookie)
    // Add the contact to the service data model's contact collection.
    context.AddToAccountCollection(account);
    // Add the account to the service data model's account collection.
    context.AddToContactCollection(contact);
    // Connect the contact to the account in the service data model.
    context.SetLink(contact, "Account", account);
    // Save the changes to Creatio in a single request.
    DataServiceResponse responces = context.SaveChanges(SaveChangesOptions.Batch);
    // Process the server's responses.
}
```

Update an object

```
public static void UpdateCreatioEntityByOdatetWcfExample()
{
    // Create the Creatio application context.
    var context = new Creatio(serverUri);
    // Define the method that adds authentication cookies upon creating a new request.
    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    // Select a collection's contact to update.
    var updateContact = context.ContactCollection.Where(c =< c.Name.Contains("Test")).First();
    // Update the contact's properties.
    updateContact.Notes = "New updated description for this contact.";
    updateContact.Phone = "123456789";
    // Save the changes to the service data model.
    context.UpdateObject(updateContact);
    // Save the changes to Creatio in a single request.
    var responces = context.SaveChanges(SaveChangesOptions.Batch);
}
```

Delete an object

```
public static void DeleteCreatioEntityByOdataWcfExample()
```

```
{
    // Create the Creatio application context.
    var context = new Creatio(serverUri);

    context.SendingRequest += new EventHandler<SendingRequestEventArgs>(OnSendingRequestCookie);
    // Select a contact collection's object to delete.
    var deleteContact = context.ContactCollection.Where(c => c.Name.Contains("Test")).First();
    // Delete the object from the service data model.
    context.DeleteObject(deleteContact);
    // Save the changes to Creatio in a single request.
    var responces = context.SaveChanges(SaveChangesOptions.Batch);
    // Process the server's responses.
}
```

# Odata web service (OData 4) API

Advanced

Depending on the request type, OData 4 protocol can return different data. Learn more about the request and response structure below.

Request structure

```
// Request string.
method Creatio_application_address/0/odata/objects_collection(object_id)/object_field?$parameter

// Request headers.
Accept: application/json
Content-Type: application/json; charset=utf-8; IEEE754Compatible=true
ForceUseSession: true
BPMCSRF: authentication_cookie_value

// Request body (for POST and PATCH requests).
{
    "field1": "value1",
    "field2": "value2",
    ...
}
```

Response structure

```
// The status code.
```

```
Status: code

// Response body (available in GET and post requests).
{
    "@odata.context": "http://Creatio_application_address/0/odata/$metadata#data_resource",
    "value": [
    {
        "object1 field1": "object1 field_value1",
        "object1 field2": "object1 field_value2",
        ...
    },
    {
        "object2 field1": "object2 field_value1",
        "object2 field2": "object2 field_value2",
        ...
    },
    ...
    ]
}
```

## Request string

`method` **required**

Creatio supports the following request methods:

- `GET` – retrieve data
- `POST` – add data
- `PATCH` – modify data
- `DELETE` – delete data

`Creatio_application_address` **required**

Creatio application URL.

`odata` **required**

OData 4 protocol's web service URL. Unmodifiable part of the request.

`objects_collection` **required**

Name of the database table (the name of the object collection). Run the authentication and execute one of the requests to retrieve the list of database tables.

JSON

```
// The result will be received in JSON.
GET Creatio_application_address/0/odata/

// Request headers.
ForceUseSession: true
BPMCSRF: authentication_cookie_value
```

XML

```
// The result will be received in XML.
GET Creatio_application_address/0/odata/$metadata

// Request headers.
ForceUseSession: true
BPMCSRF: authentication_cookie_value
```

[SysSchema] table data

```
// The result will be received from the [SysSchema] database table in JSON.
GET Creatio_application_address/0/odata/SysSchema?$filter=ManagerName eq 'EntitySchemaManager

// Request headers.
ForceUseSession: true
BPMCSRF: authentication_cookie_value
```

---

`object_id` **optional**

Identifier of the database table record string (the identifier of the collection object instance).

---

`object_field` **optional**

Database table record field (the field of the collection object instance).

---

`parameters` **optional**

Optional OData 4 parameters you can use in the `GET` Creatio request string. Use the `?` Operator to specify the parameters. Add the parameter name after the `$` operator. Use the `&` operator to use two or more parameters.

### Available parameters

| | | |
|---|---|---|
| `$value` | | The field value. |
| `$count` | `$count=true` | The number of elements in the selection. |
| `$skip` | `$skip=n` | The first n elements that must be excluded from the selection. |
| `$top` | `$top=n` | The first n elements that must be included in the selection. |
| `$select` | `$select=field1,field2,...` | A set of fields that must be included in the selection. |
| `$orderby` | `$orderby=field asc` or `$orderby=field desc` | How to sort the field values in the selection. |
| `$expand` | `$expand=field1,field2,...` | Extension of the connected fields. |
| `$filter` | `$filter=field template 'field_value'` | How to filter the fields in the selection. |

## Request headers

`Accept` `application/json` **required**

Data type to expect in the server response. Optional for `GET` requests.

`Content-Type` `application/json; charset=utf-8; IEEE754Compatible=true` **required**

Encoding and type of the resource passed in the request body. Optional for `GET` requests.

`ForceUseSession` `true` **required**

`ForceUseSession` header forces the use of an existing session.

`BPMCSRF` `authentication_cookie_value` **required**

Authentication cookie.

## Request body

`field1, field2,...` **required**

`field1, field2, ...` **required**

The field names passed in the request body.

---

`value1, value2, ...` **required**

Values of the `field1, field2, …` fields passed in the request body.

# HTTP status code

---

`code`

Response status code.

Available status codes

| | |
|---|---|
| **200 OK** | A request that does not create a resource was completed successfully, and the resource's value does not equal 0. In this case, the response body should contain the value of the resource specified in the request URL. The information in the response depends on the request method:<br>`GET` – the resource was found and passed in the response body.<br>`POST` – the resource with the description of server actions caused by the request was passed in the response body. |
| **201 Created** | A request that creates a resource successfully. The response body should contain the created resource. Used for `POST` requests that create a collection, create a multimedia object (e. g., a photo), or call an action through import. |
| **202 Accepted** | Data request processing has started but has not finished yet. There is no guarantee that the request will be completed successfully (asynchronous request processing). |
| **204 No content** | The request was processed successfully, but there is no need to return any data. The value of the requested resource is 0. The response will pass only the headers, the response body should be empty. |
| **3xx Redirection** | Redirection means the client must take further actions to execute the request. The response should contain the `Location` header with the URL that can be used to retrieve the result. The response can also contain the `Retry-After` header that displays time, in seconds. The time specifies how long the client can wait before executing another request to the resource in the `Location` header. |

| | |
|---|---|
| **304 Not modified** | The client executes a `GET` request with the `If-None-Match` header, and the content remains unchanged. The response should not contain any other headers. |
| **403 Forbidden** | The request is correct, but the server refused to authorize it. This means the client lacks permissions to work with the resource. This may be caused by an invalid `BPMCSRF` cookie. |
| **404 Not Found** | The server cannot find the resource specified in the URL. The response body may contain additional information. |
| **405 Method Not Allowed** | The resource specified in the request URL does not support the specified request method. The response should contain the `Allow` header with the list of request methods the resource supports. |
| **406 Not Acceptable** | The resource specified in the request URL does not have any current view that is acceptable for the client as per `Accept`, `Accept-Charset`, and `Accept-Language` request headers. The service does not provide a default view. |
| **410 Gone** | The requested resource is no longer available. The resource had used the specified URL but was deleted and is no longer available. |
| **412 Prediction Failed** | The client specified a request header condition the resource cannot process. |
| **424 Failed Dependency** | The current request cannot be processed because the requested action depends on another action that could not be executed. The request has not been executed due to dependency failure. |
| **501 Not Implemented** | The client is using a request method that is not implemented in OData 4 protocol and cannot be processed. The response body should contain the description of the unimplemented functionality. |

## Response body

`@odata.context`

Information about the type of the returned data. Besides the data source path, the `data_resource` element can contain the `$entity` parameter. This parameter indicates that the response returned a single instance of the collection object. Available only for `GET` and `POST` requests.

`value`

Contains the object collection. Not available if the response contains a single collection object instance. Available only for `GET` requests.

---

`[]`

Object collection. Available only for `GET` requests.

---

`{}`

Collection object instances. Available only for `GET` and `POST` requests.

---

`object1 field1, object1 field2, ..., object2 field1, object2 field2, ...`

The names of the `field1, field2, …` fields in the `object1, object2, …` collection object instances. Available only for `GET` and `POST` requests.

---

`object1 field_value1, object1 field_value2, ..., object2 field_value1, object2 field_value2, ...`

The values of the `field1, field2, …` fields in the `object1, object2, …` collection object instances. Available only for `GET` and `POST` requests.

# EntityDataService.svc web service (OData 3) API

Advanced

Depending on the request type, OData 3 protocol can return different data. Learn more about the request and response structure below.

Request structure

```
// Request string.
method Creatio_application_address/0/ServiceModel/EntityDataService.svc/objects_collectionCollec

// Request headers.
Accept: application/atom+xml; type=entry
Content-Type: application/json; odata=verbose
ForceUseSession: true
BPMCSRF: authentication_cookie_value

// Request body (for POST and PATCH requests).
```

```
{
    "field1": "value1",
    "field2": "value2",
    ...
}
```

Response structure

```
// The status code.
Status: code

// Response body (available in GET and POST requests).
<?xml version="1.0" encoding="utf-8"?>
<feed xml:base="http://mycreatio.com/0/ServiceModel/EntityDataService.svc/" xmlns="http://www.w3
    <id>http://mycreatio.com/0/ServiceModel/EntityDataService.svc/data_resource</id>
    <title type="text">data_resource</title>
    <updated>date and time of request</updated>
    <link rel="self" title="data_resource" href="data_resource" />
    <entry>
        metadata_data
        <content type="application/xml">
            <m:properties>
                <d:object1 field1>object1 field_value1</d:object1 field1>
                <d:object1 field2>object1 field_value2</d:object1 field2>
                ...
            </m:properties>
        </content>
    </entry>
    <entry>
        metadata_data
        <content type="application/xml">
            <m:properties>
                <d:object2 field1>object2 field_value1</d:object2 field1>
                <d:object2 field2>object2 field_value2</d:object2 field2>
                ...
            </m:properties>
        </content>
    </entry>
    ...
</feed>
```

# Request string

`method` **required**

Creatio supports the following request methods:

- `GET` – retrieve data
- `POST` – add data
- `PATCH` – modify data
- `DELETE` – delete data

---

`Creatio_application_address` **required**

Creatio application URL.

---

`ServiceModel` **required**

OData 3 protocol's web service URL. Unmodifiable part of the request.

---

`EntityDataService.svc` **required**

OData 3 protocol's web service URL. Unmodifiable part of the request.

---

`objects_collectionCollection` **required**

Name of the database table (name of the object collection). When using the OData 3 protocol, add `Collection` to the first name of the object collection in the request string (e.g., `ContactCollection`). Run a query to receive the list of database tables.

MySQL

```
SELECT * FROM INFORMATION_SCHEMA.TABLES
```

Oracle

```
SELECT * FROM ALL_TABLES
```

PostgreSQL

```
SELECT table_name FROM information_schema.tables
```

`guid'object_id'` **optional**

The identifier of the database table record string (identifier of the collection object instance). For example, `guid'00000000-0000-0000-0000-000000000000'`).

`object_field` **optional**

The database table record field (field of the collection object instance).

`parameters` **optional**

Optional OData 3 parameters you can use in the `GET` Creatio request string. Use the `?` operator to specify the parameters. Add the parameter name after the `$` operator. Use the `&` operator to use two or more parameters.

### Available parameters

| | | |
|---|---|---|
| $value | | The field value. |
| $count | `$count=true` | The number of elements in the selection. |
| $skip | `$skip=n` | The first n elements that must be excluded from the selection. |
| $top | `$top=n` | The first n elements that must be included in the selection. |
| $select | `$select=field1,field2,...` | The set of fields that must be included in the selection. |
| $orderby | `$orderby=field asc` or `$orderby=field desc` | How to sort the field values in the selection. |
| $expand | `$expand=field1,field2,...` | Extension of the connected fields. |
| $filter | `$filter=field template 'field_value'` | How to filter the fields in the selection. |

## Request headers

`Accept` `application/atom+xml; type=entry` **required**

Data type to expect in the server response. The server returns the response in XML. Optional for `GET` requests.

`Content-Type` `application/json; odata=verbose` **required**

Ecoding and type of the resource passed in the request body. Optional for `GET` requests.

`ForceUseSession` `true` **required**

The `ForceUseSession` header forces the use of an existing session. You do not need to use `AuthService.svc` in your request to the authentication service.

`BPMCSRF` `authentication_cookie_value` **required**

Authentication cookie.

## Request body

`field1, field2, ...` **required**

The names of the fields passed in the request body.

`value1, value2, ...` **required**

The values of the `field1, field2, …` fields passed in the request body.

## HTTP status code

`code`

Response status code.

Available status codes

| | |
|---|---|
| **200 OK** | `GET`, `PUT`, `MERGE`, or `PATCH` request was completed successfully. The response body should contain the value of the object or properly specified in the request URL. |
| **201 Created** | `POST` request created an object or a link successfully. The response body should contain the updated object. |
| **202 Accepted** | Processing of the data update request has started but has not finished yet. The response body should contain the `Location` header and the `Retry-After` header. The response body should be empty. The server should return the 303 response code with the `Location` header that contains the final URL that can be used to retrieve the result. The body and the header of the final URL should be formatted similar to the initial data update request. |
| **204 No content** | Data update request. The value of the requested resource is 0. The response body should be empty. |
| **3xx Redirection** | Data update request. The redirection means the client must take further actions to execute the request. The response should contain the `Location` header with the URL that can be used to retrieve the result. |
| **4xx Client Error** | Incorrect requests. The server returns this code in response to client errors and requests to non-existent resources, such as entities, entity collections, or properties. If the response body is defined for the error code, the error body will be as defined for the corresponding format. |
| **404 Not Found** | The object or collection specified in the URL does not exist. The response body should be empty. |

## Response body

`entry`

The instance of the collection object.

`metadata_data`

The metadata of the collection object instance.

`object1 field1, object1 field2, ..., object2 field1, object2 field2, ...`

The names of the `field1, field2, …` fields in the `object1, object2, …` collection object instances.

```
object1 field_value1, object1 field_value2, ..., object2 field_value1, object2 field_value2, ...
```

The values of the `field1, field2, …` fields in the `object1, object2, …` collection object instances.