centerbase

Table of Contents

[Centerbase Developer's API 1](#_Toc109229050)

[1. Introduction 3](#_Toc109229051)

[1.1 Using Fiddler 3](#_Toc109229052)

[1.2 JSON Types and Syntax 4](#_Toc109229053)

[1.3 Using a Service Proxy in .NET 5](#_Toc109229054)

[2. Web API 7](#_Toc109229055)

[2.1 Login 7](#_Toc109229056)

[2.2. Company/Contact Exists 8](#_Toc109229057)

[2.3. Create Contact/Company 9](#_Toc109229058)

[2.4. Update Item 10](#_Toc109229059)

[2.5. Create Note 11](#_Toc109229060)

[2.6. Link Items 12](#_Toc109229061)

[2.7. Unlink Items 13](#_Toc109229062)

[2.8. Create Activity 14](#_Toc109229063)

[2.9. Get Users List 15](#_Toc109229064)

[2.10. Get Query Items 16](#_Toc109229065)

[3. Centerbase Command-Line 17](#_Toc109229066)

[Appendix A. Objects 19](#_Toc109229067)

[A.1 Field-Info Object 19](#_Toc109229068)

[A.2 Item Object 19](#_Toc109229069)

[A.3 Field-Data Object 19](#_Toc109229070)

[Appendix B. Field Identifiers 20](#_Toc109229071)

[B.1 System Fields 20](#_Toc109229072)

[B.2 Contact Fields 20](#_Toc109229073)

[B.3 Company Fields 21](#_Toc109229074)

[B.4 Activity Fields 21](#_Toc109229075)

[B.5 Note Fields 21](#_Toc109229076)

[B.6 User Fields 21](#_Toc109229077)

[B.7 Custom Fields 22](#_Toc109229078)

[Appendix C. Item Types 23](#_Toc109229079)

[Appendix D. Error Codes 23](#_Toc109229080)

# 1. Introduction

This document outlines and describes the services available through the Centerbase Web API and Centerbase command-line. Let's start by saying that the API is implemented as a REST web service that uses JSON as its data interchange format. REST is a web service model that sends and receives information through HTTP without building the SOAP envelope used by traditional web services. This streamlines the amount of data sent and received, making the communication process very efficient.

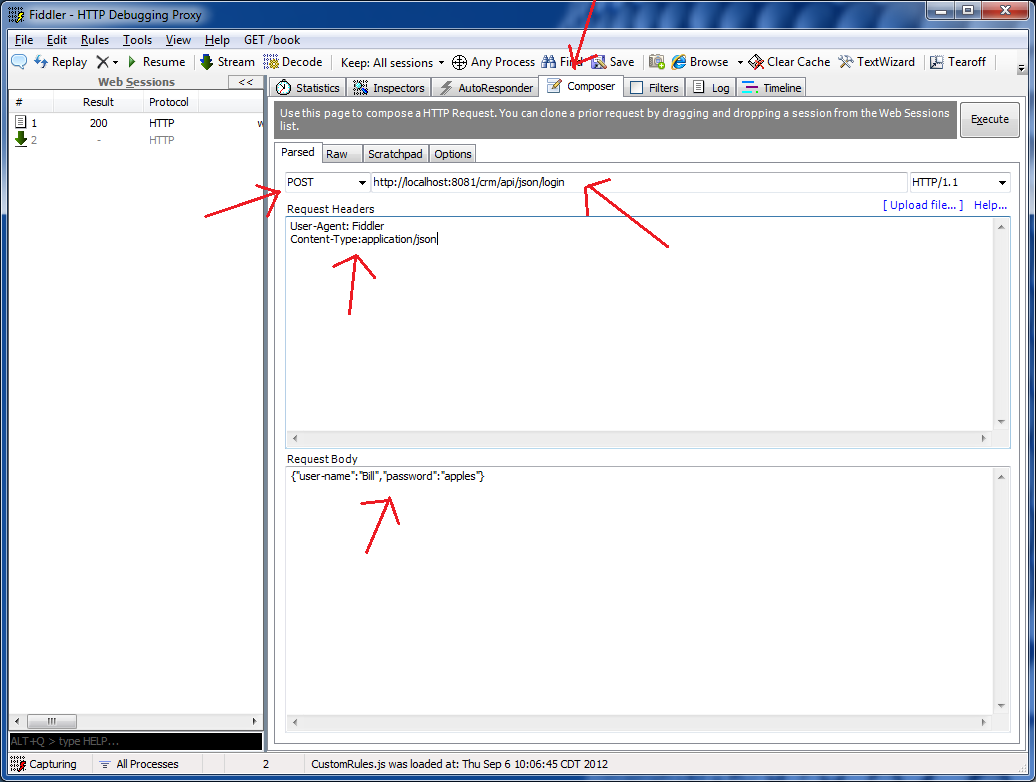
## 1.1 Using Fiddler

Fiddler is a web debugging proxy that can be downloaded from the web for free and it is a very popular tool that can be used to test the Centerbase Web API service. To do that, download and install Fiddler, launch it and follow these steps:

* Go to the **Composer** tab and enter the URI of the resource you want to work with, i.e. *http:\\<yourhost>\api\json\login*
* Select the request method, i.e. POST.
* In the **Request Headers** section, specify the content-type, enter: *Content-Type:application/json*
* In the **Request Body** section, enter the appropriate JSON fragment. In the case of the login resource, it should look something similar to this:

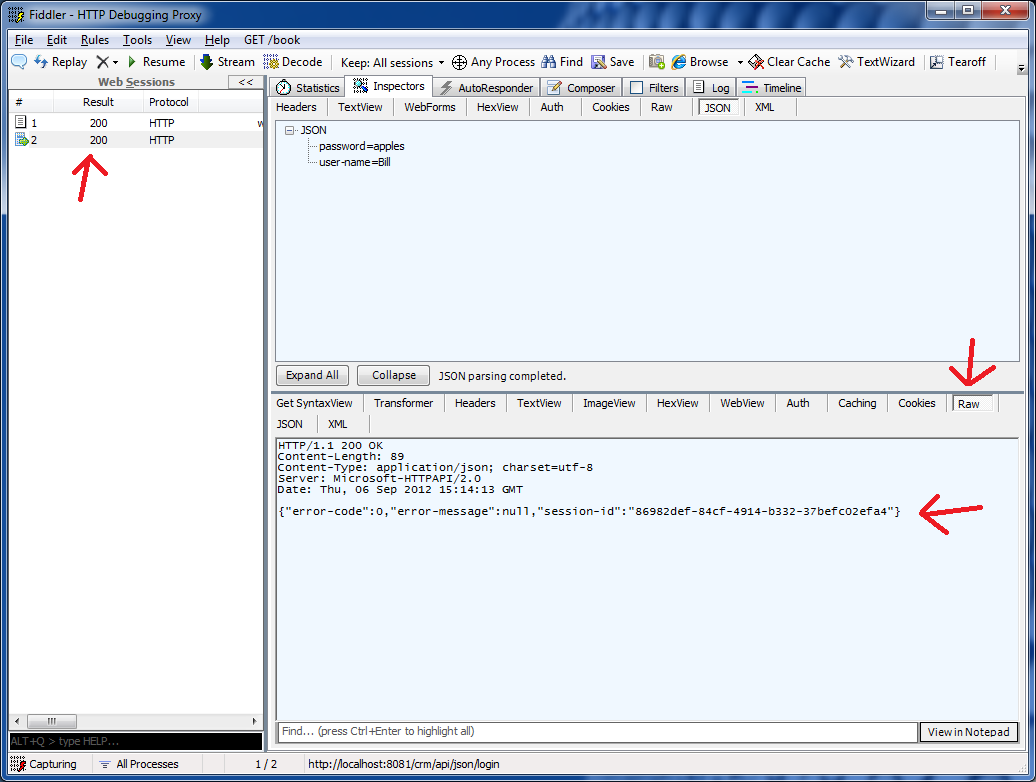
|  |
| --- |
| {"user-name":"Bill","password":"Cowboys"} |

* Click on the **Execute** button.



*Entering Request*

To inspect the results, on the left panel, click on the response you are interested in and select the **Raw** or **JSON** tab. The status code or result of the response should be 200 (OK).



*Inspecting Response*

## 1.2 JSON Types and Syntax

JSON's basic types are:

* **Number** (double precision floating-point format)
* **String** (double-quoted, with backslash escaping)
* **Boolean** (true or false)
* **Array** (An ordered sequence of values, comma-separated and enclosed in square brackets; the values do not need to be of the same type)
* **Object** (an unordered collection of key-value pairs with a colon ':' separating the key and the value, comma-separated and enclosed in curly brackets)
* **null** (empty)

Centerbase passes dates as strings and dates should always be in Coordinated Universal Time (UTC) using the sortable date/time pattern.

In the .NET framework use 's' as the date format specifier:

|  |
| --- |
| string sortableDateFormat = DateTime.UTCNow().ToString("s"); |

## 1.3 Using a Service Proxy in .NET

To call the Centerbase Service or Web API using the .NET framework (3.5 or higher), include the service proxy file (**CBServiceClient.cs** or **CBServiceClient.vb)** in your project and add the following references:

System.ServiceModel

System.ServiceMode.Web

System.Runtime.Serialization

Modify your application configuration file (or Web.config) to include the **<system.serviceModel>** section as shown below:

|  |
| --- |
| <?xml version="1.0" encoding="utf-8" ?>  <configuration>    <!-- More... -->    <system.serviceModel>  <client>  <endpoint address="http://***<yourhost>***/api"  behaviorConfiguration="webhttp"  binding="webHttpBinding"  contract="Centerbase.ICBAPIService"  name="WebHttpBinding\_ICBApiService" />  </client>  <behaviors>  <endpointBehaviors>  <behavior name="webhttp">  <webHttp/>  </behavior>  </endpointBehaviors>  </behaviors>  </system.serviceModel>    <!-- More... -->    </configuration> |

*Configuration File*

Don’t forget to update ***<yourhost>*** with the appropriate information.

To use the proxy, instantiate an object of the **CBAPIServiceClient** class and start using it. See example below:

C#

|  |
| --- |
| using Centerbase;  class Program  {  static void Main(string[] args)  {  // Instantiate proxy.  CBAPIServiceClient proxy = new CBAPIServiceClient();  // Create and fill request.  CBAPIRequestLogin request = new CBAPIRequestLogin();  request.UserName = "Bill";  request.Password = "Cowboys";  // Call service.  CBAPIResponseLogin response = proxy.Login(request);  // Check response.  if (response.SessionID != Guid.Empty)  {  Console.WriteLine("You have signed in successfully!");  }  // Always close the client.  proxy.Close();  }  } |
|  |
|  |

VB

|  |
| --- |
| Imports Centerbase  Class Program  Private Shared Sub Main(ByVal args As String())  ' Instantiate proxy.  Dim proxy As New CBAPIServiceClient()  ' Create and fill request.  Dim request As New CBAPIRequestLogin()  request.UserName = "Bill"  request.Password = "Cowboys"  ' Call service.  Dim response As CBAPIResponseLogin = proxy.Login(request)  ' Check response.  If response.SessionID <> Guid.Empty Then  Console.WriteLine("You have signed in successfully!")  End If  ' Always close the client.  proxy.Close();  End Sub  End Class |
|  |
|  |

# 2. Web API

## 2.1 Login

|  |  |
| --- | --- |
| **Description**: | Signs into Centerbase using your Centerbase user name and password. |
| **URL:** | http://<yourhost>/api/json/login |
| **Method:** | POST |
| **Content-Type**: | application/json |

**Request Body:**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *user-name* | Your regular Centerbase user name. |
| *Password* | Your Centerbase password. |

Request-body sample:

|  |
| --- |
| {"user-name":"Bill","password":"Cowboys"} |

**Response:**

|  |  |
| --- | --- |
| ***Output*** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *session-id* | If the credentials are valid, it returns a session ID. This session ID should be used in subsequent API calls. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null,"session-id":"f3cc9c11-f4d3-4ad8-a605-6399f7432cc6"} |

## 2.2. Company/Contact Exists

|  |  |
| --- | --- |
| **Description**: | This service receives the criteria used to determine what a match is and the values needed to perform this verification. It returns a list of matches, that includes the name and ID of the item and a list of values that can be displayed for verification purposes. |
| **URL:** | http://<yourhost>/api/json/item/exists |
| **Method:** | PUT |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *item-type* | Requested item type: "CONTACT" or "COMPANY". |
| *Criteria* | Array of Field-Data objects (See Appendix A).  This contains the fields and values used to find existing Contacts/Companies. The value field in the Field-Data object contains the specific pattern to match either an exact match or a string pattern using T-SQL LIKE wildcards. |
| *requested-fields* | Array of strings with the field IDs to be returned. (See appendix B.)  This specifies the fields that you want to retrieve. |
| *session-id* | ID returned at login time. |

Request-body sample:

|  |
| --- |
| {"criteria":[{"field-id":"FULLNAME","value":"John Smith"},{"field-id":"E-MAIL","value":"jsmith@acme.com"}],"item-type":"CONTACT","requested-fields":["E-MAIL","WORK-PHONE-NBR"],"session-id":"8e4e406a-0d6e-4715-8661-2f0f14cd1035"} |

**Response**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *field-info* | Array of field information (See "Field-Info Object" in Appendix A). |
| *items* | Array of items matching the criteria (See Appendix A). |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null,"field-info":[{"data-type":"String","field-id":"NAME","label":"Contact name"},{"data-type":"E-mail","field-id":"E-MAIL","label":"E-mail address"}],"items":[{"field-data":[{"field-id":"E-MAIL","value":"jsmith@acme.com"},{"field-id":"WORK-PHONE-NBR","value":"214-671-0209"}],"item-id":"35d13114-08c6-4e84-a9c8-333fe91d1042","name":"John Smith"}]} |

## 2.3. Create Contact/Company

|  |  |
| --- | --- |
| **Description**: | This service creates a Contact/Company. In the case of a Contact, it automatically parses the contact's name (FULLNAME field) into the appropriate fields, and returns the new Centerbase identifier. |
| **URL:** | http://<yourhost>/api/json/item |
| **Method:** | POST |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *field-data* | Array with Field-Data objects  (See Field Data Object in Appendix A).  Contains the field values you want to set on the new Contact/Company. |
| *item-type* | i.e. "CONTACT" or "COMPANY" |
| *session-id* | ID returned at login time |

Body-request sample:

|  |
| --- |
| {"field-data":[{"field-id":"NAME","value":"Centerbase, Inc."},{"field-id":"PHONE-NBR","value":"(214) 987-9070"},{"field-id":"ADDRESS-CITY","value":"Dallas"}],"item-type":"COMPANY","session-id":"d0fe4ea2-a47a-4b00-9ac7-6dd109faaf11"} |

**Response**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *item-id* | Unique item identifier of the newly created item. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null,"item-id":"64919064-ca97-4c7b-ad71-078d5346d34c"} |

## 2.4. Update Item

|  |  |
| --- | --- |
| **Description**: | Receives a Centerbase identifier and the new field values needed for the update. |
| **URL:** | http://<yourhost>/api/json/item |
| **Method:** | PUT |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *item-id* | The ID of the item to be updated (GUID).  (This Centerbase ID could be for a Contact, Company, Activity or Note). |
| *field-data* | Array with Field-Data objects (See Appendix A).  Contains the field values to be set. |
| *session-id* | ID returned at login time |

Request-body sample:

|  |
| --- |
| {"field-data":[{"field-id":"ADDRESS-CITY","value":"San Diego"},{"field-id":"ADDRESS-STATE","value":"CA"}],"item-id":"c767843c-83f4-4a26-b1e1-cfc4aeebb9cf","session-id":null} |

**Response:**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null} |

## 2.5. Create Note

|  |  |
| --- | --- |
| **Description**: | Creates a note in Centerbase and it links it to an existing item. |
| **URL:** | http://<yourhost>/api/json/note |
| **Method:** | POST |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *parent-id* | The ID of the item we are adding the note to (GUID). |
| *note-text* | The text of the note to create. |
| *session-id* | ID returned at login time. |

Body-request sample:

|  |
| --- |
| {"note-text":"This is a sample of a very short note.","parent-id":"0be48081-e371-45a8-9068-e96a8c77267e","session-id":"786d5ab1-8467-4b93-aa9b-696bc8818964"} |

**Response:**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *item-id* | Newly created Note identifier. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null,"item-id":"78296de2-3dad-413e-9506-9237a45d1c54"} |

## 2.6. Link Items

|  |  |
| --- | --- |
| **Description**: | Receives a collection of Centerbase identifiers and links them all to another item. |
| **URL:** | http://<yourhost>/api/json/item/link |
| **Method:** | POST |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *item-list* | Array of identifiers (GUIDs) of the items we want to link. |
| *parent-id* | Identifier of the item we want to link to (GUID). |
| *session-id* | ID returned at login time |

Body-request sample:

|  |
| --- |
| {"item-list":["082c3026-b32f-4d59-826c-5bdb503186e0","c5c3eeac-f676-4475-9f24-89eead46e666","61ad5b6f-b905-476c-866f-510a609d3a6a"],"parent-id":"e2de7ef0-6690-4dc8-82e7-80f6354de131","session-id":"4932f929-eafd-45c3-aed1-f9396498312e"} |

**Response:**

|  |  |
| --- | --- |
| ***Output*** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null} |

## 2.7. Unlink Items

|  |  |
| --- | --- |
| **Description**: | Receives a collection of Centerbase identifiers and removes their link from another item. |
| **URL:** | http://<yourhost>/api/json/item/unlink |
| **Method:** | PUT |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *item-list* | Array of identifiers (GUIDs) of the items we want to unlink from the parent. |
| *parent-id* | Identifier of the item we want to unlink from (GUID). |
| *session-id* | ID returned at login time |

Request-body sample:

|  |
| --- |
| {"item-list":["082c3026-b32f-4d59-826c-5bdb503186e0","c5c3eeac-f676-4475-9f24-89eead46e666","61ad5b6f-b905-476c-866f-510a609d3a6a"],"parent-id":"e2de7ef0-6690-4dc8-82e7-80f6354de131","session-id":"4932f929-eafd-45c3-aed1-f9396498312e"} |

**Response:**

|  |  |
| --- | --- |
| ***Output*** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null} |

## 2.8. Create Activity

|  |  |
| --- | --- |
| **Description**: | Receives the field values needed to create a new task or appointment. |
| **URL:** | http://<yourhost>/api/json/activity |
| **Method:** | POST |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *all-day-event* | If true, the start/end date should not have a time associated with it (true or false). |
| *end-date* | Date must be sent in Universal Time and in sortable format ('s'). |
| *location* | Could be any text value. |
| *schedule-by* | Centerbase Identifier. Must be the ID (GUID) of a valid Centerbase user. If not supplied it defaults to the logged in user. |
| *schedule-for* | Array of Centerbase identifiers. Can be the ID of Company, Contact or User. |
| *session-id* | ID returned at login time. |
| *start-date* | Date must be sent in Universal Time and in sortable format ('s') |
| *subject* | Cold be any text value |
| *type* | "TASK" or "APPOINTMENT"  If it is not specified it defaults to “APPOINTMENT” |

Request-body sample:

|  |
| --- |
| {"all-day-event":false,"end-date":"2012-09-05T19:40:57","location":"Conference room","schedule-by":"93490ff8-8f7d-4833-9239-51534919fddf","schedule-for":["ddf3d2c5-b0b9-4a74-910e-d2a489edac9f","0f813238-477f-4953-9b77-2a61818001c7"],"session-id":"ceed5835-2993-439b-be67-b572e44deb41","start-date":"2012-09-05T18:40:57","subject":"Planning meeting","type":"APPOINTMENT"} |

**Response:**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *activity-id* | Newly created activity ID |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |

Response sample:

|  |
| --- |
| {"activity-id":"cf8ac222-ea7c-44b6-86b7-a0d43633e4d4","error-code":0,"error-message":null} |

## 2.9. Get Users List

|  |  |
| --- | --- |
| **Description**: | Returns the list of users and their properties. |
| **URL:** | http://<yourhost>/api/json/user |
| **Method:** | PUT |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *active-status* | Specifies the type of users that are returned. "ACTIVE", "INACTIVE" or "BOTH". |
| *field-ids* | Array of field IDs (See Appendix B). Specifies the fields that are returned. |
| *session-id* | ID returned at login time. |

Body-request sample:

|  |
| --- |
| {"active-status":"ACTIVE","field-ids":["NAME"],"session-id":"7402b3e5-a7f3-4abd-a72f-b8aa73447478"} |

**Response:**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *field-info* | Array of field information (See "Field-Info Object" in Appendix A). |
| *users* | Array of Item Objects (See Appendix A) |

Response sample:

|  |
| --- |
| {"error-code":0,"error-message":null,"field-info":[{"data-type":"String","field-id":"NAME","label":"User name"}],"users":[{"field-data":[{"field-id":"FULLNAME","value":"John Smith"},{"field-id":"FULLNAME","value":"Bill Williams"},{"field-id":"FULLNAME","value":"Michael Johnson"}],"item-id":"d86e515f-9135-49e9-90e0-09f51ef97ced","name":null}]} |

## 2.10. Get Query Items

|  |  |
| --- | --- |
| **Description**: | Returns the list of items and their properties. |
| **URL:** | http://<yourhost>/api/json/query |
| **Method:** | PUT |
| **Content-Type**: | application/json |

**Request Body**:

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *item-type* | Requested item type (1001, 32, etc) |
| *Criteria* | Array of Field-Data objects (See Appendix A).  This contains the fields and values used to find existing Contacts/Companies. The value field in the Field-Data object contains the specific pattern to match either an exact match or a string pattern using T-SQL LIKE wildcards. |
| *requested-fields* | Array of strings with the field IDs to be returned. (See appendix B.)  This specifies the fields that you want to retrieve. |
| *session-id* | ID returned at login time. |

Body-request sample:

|  |
| --- |
| {  "item-type" : "1001",  "criteria" : "",  "requested-fields" : "",  "session-id" : "5385d330-d155-437d-a0ba-da0948a43f8a"  } |

**Response:**

|  |  |
| --- | --- |
| **Output** | **Description** |
| *error-code* | See Appendix D. |
| *error-message* | Description of the error condition. |
| *results-id* | ID of the results in the DB |
| *results-set* | Array of Item Objects (See Appendix A) |

Response sample:

|  |
| --- |
| {  "error-code": 0,  "error-message": null,  "results-id": "2edb1053-564e-41eb-af2a-f6be6f9019e4",  "results-set": [  {  "AdditionalProperties": [],  "Fields": [],  "Inactive": false,  "Name": "",  "ObjectId": "50398cbf-d835-462b-bbc2-e64331319df6",  "ObjectType": 1001,  "ParentObjectId": "00000000-0000-0000-0000-000000000000",  "SubType": -1  },  ]  } |

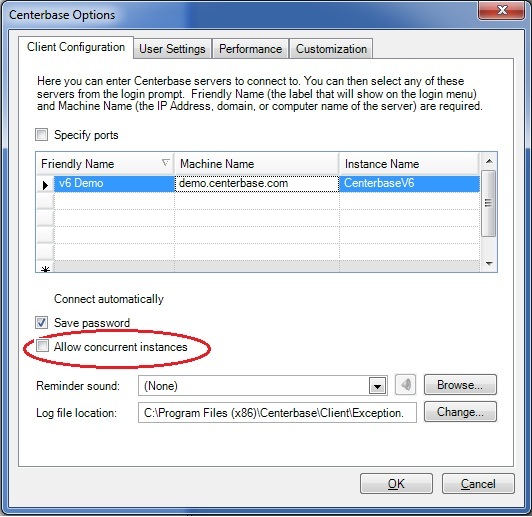
# 3. Centerbase Command-Line

It displays a Company or Contact in the Centerbase client. Centerbase is instructed to open the specified Company or Contact given its Centerbase item identifier.

Syntax:

**CENTERBASE.EXE /OPEN** <Centerbase item identifier>

To get the best results for the command line, it is recommended that Centerbase be run in Single Instance mode. To do this, open Centerbase and go to Centerbase Options. Un-check the box for “Allow concurrent instances”:

****

*Centerbase Options*

Then make sure you close ALL copies of Centerbase that are open on that machine. The next time Centerbase is run (whether from the command line or in general) it will be in Single Instance mode. This means every time you send the /open command line, it will open that item in the Centerbase that is already open instead of opening a new copy of Centerbase.

NOTE: This is a local per-machine setting. This means each computer you have Centerbase installed on that you want to use this will need to un-check the option and close/re-open Centerbase.

The command line also works if Centerbase is not open. Simply run the /open command line and Centerbase will pop up and prompt the user to log in. Once they are logged in and Centerbase is open, it will open the item that you specified. All further command line actions will be sent to this newly opened copy of Centerbase.

# Appendix A. Objects

## A.1 Field-Info Object

|  |  |
| --- | --- |
| ***Key*** | **Value** |
| *field-id* | Field identifier i.e. "LAST-NAME" (See appendix B). |
| *label* | This is the label that is used by Centerbase when displaying the field on its forms. |
| *data-type* | String, Number, Lookup, Phone, Address, E-mail Address, Memo, YesNo, Date, DateTime. |

## A.2 Item Object

|  |  |
| --- | --- |
| ***Key*** | **Value** |
| *item-id* | Unique item identifier. |
| *name* | Item name. |
| *field-data* | Array of Field-Data objects (See below). |

## A.3 Field-Data Object

|  |  |
| --- | --- |
| ***Key*** | **Value** |
| *field-id* | Field identifier i.e. "LAST-NAME" (See appendix B). |
| *value* | Text value. |

# Appendix B. Field Identifiers

Note 1: Any field marked with a star '**\***' is read-only (cannot be edited).

Note 2: All field identifiers are case-sensitive.

## B.1 System Fields

|  |
| --- |
| **FIELD ID** |
| ACTIVE-STATUS |
| CLASS |
| CREATION-DATE \* |
| CREATOR \* |
| NAME |

## B.2 Contact Fields

|  |
| --- |
| **FIELD ID** |
| FULLNAME |
| FIRST-NAME |
| MIDDLE-NAME |
| LAST-NAME |
| HOME-PHONE-NBR |
| MOBILE-PHONE-NBR |
| WORK-PHONE-NBR |
| E-MAIL-ADDRESS |
| ADDRESS-LINE1 |
| ADDRESS-LINE2 |
| ADDRESS-LINE3 |
| ADDRESS-CITY |
| ADDRESS-COUNTY |
| ADDRESS-STATE |
| ADDRESS-POSTAL-CODE |
| ADDRESS-COUNTRY |
| COMPANY-ADDRESS-CITY \* |
| COMPANY-ADDRESS-COUNTRY \* |
| COMPANY-ADDRESS-COUNTY \* |
| COMPANY-ADDRESS-LINE1 \* |
| COMPANY-ADDRESS-LINE2 \* |
| COMPANY-ADDRESS-LINE3 \* |
| COMPANY-ADDRESS-POSTAL-CODE \* |
| COMPANY-ADDRESS-STATE \* |
| COMPANY-ID |
| COMPANY-NAME \* |
| COMPANY-PHONE \* |
| COMPANY-URL \* |
| System fields (see above) |

## B.3 Company Fields

|  |
| --- |
| **FIELD ID** |
| ADDRESS-LINE1 |
| ADDRESS-LINE2 |
| ADDRESS-LINE3 |
| ADDRESS-CITY |
| ADDRESS-COUNTY |
| ADDRESS-STATE |
| ADDRESS-POSTAL-CODE |
| ADDRESS-COUNTRY |
| PHONE-NBR |
| FAX-NBR |
| URL |
| System fields (see above) |

## B.4 Activity Fields

|  |
| --- |
| **FIELD ID** |
| ALL-DAY-EVENT |
| DESCRIPTION |
| END-DATE |
| LOCATION |
| SCHEDULED-BY |
| START-DATE |
| SUBJECT |
| TYPE |
| System fields (see above) |

## B.5 Note Fields

|  |
| --- |
| **FIELD ID** |
| DESCRIPTION |
| System fields (See above) |

## B.6 User Fields

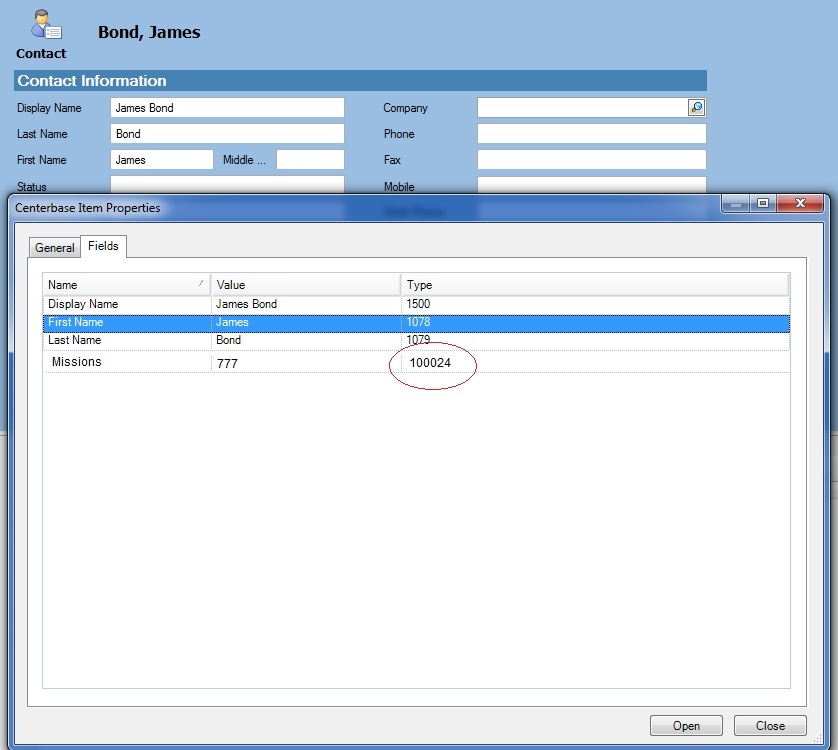
|  |
| --- |
| **FIELD ID** |
| ADMIN |
| System fields (See above) |

## B.7 Custom Fields

Custom fields do not have a pre-defined field identifier, for that reason they are identified using the following format:

*{item-numeric-id}:{field-numeric-id}*

For instance it could be "4:100024", where 4 is the code for the Contact item and 100024 is the custom-field numeric identifier. To find out the numeric field identifier of a custom field, right-click on the Centerbase form of the item type you are interested and from the context-menu select **Properties,** then click onthe **Fields** tab, the last column displays the number you are looking for, see picture below.



*Finding numeric field identifier*

|  |  |
| --- | --- |
| **Item** | **Numeric ID** |
| Contacts | 4 |
| Companies | 3 |
| Activities | 2 |
| Notes | 14 |

*Numeric item identifiers*

# Appendix C. Item Types

|  |
| --- |
| **ITEM TYPE** |
| CONTACT |
| COMPANY |
| USER |
| ACTIVITY |
| NOTE |

# Appendix D. Error Codes

|  |  |  |
| --- | --- | --- |
| **Code** | **Name** | **Description** |
| 32 | Validation exception. | Occurs when one or more input parameters are not valid. |
| 42 | Security exception. | Occurs when the current user does not have rights to create, delete, edit, or view an item. |
| 52 | Not logged in. | Occurs when the user has been logged out and his/her session ID is no longer valid. |
| 62 | Invalid login. | Occurs when the user-name/password combination given at login is invalid. |
| -1 | Operation failed. | Occurs when any other error condition comes up that does not have a standard error code. |