**INLAND REVENUE
AUTHORITY
OF SINGAPORE**

IRAS API SERVICES
INTERFACE SPECIFICATIONS

Submission of Donation Records (Corppass)

Last updated on : 4 May 2023

Version No: 1.6

Version Control

Version History

Revision Date	Version Number	Change Log Summary
04/06/2021	1.0	Creation of Document
18/08/2021	1.1	Updated Parameter Label Name in Section 4.2. Request Payload
08/09/2021	1.2	Updated Parameter Label Name, description and Sample JSON Request Payload in Section 4.2
18/11/2021	1.3	Updated Description for batchIndicator and donationAmount in Section 4.2 Request Payload
23/11/2021	1.4	Updated Sample JSON request payload in Section 4.2
01/03/2023	1.5	a. Updated hyperlink to Donation API Test Guide in Section 3.4 b. Updated Description for batchIndicator in Section 4.2 Request Payload c. Added paragraph on API submission behaviour under Section 3.6 Common Interface Information
04/05/2023	1.6	a. Updated Common Interface Information for acceptable characters in Section 3.6 b. Updated Endpoint for API Service in Section 3

Table of Contents

1. Introduction

The Inland Revenue Authority of Singapore (IRAS) provides application programming interface (API) services to allow application developers to submit and retrieve tax related matters using HTTP requests. Most of the APIs will be in the form of a JSON web service which reduces client/server coupling and thus enabling easier integration between IRAS' services with external developers.

While some services require a simple GET, others may be secured and require credentials that can be passed via HTTP header parameters which are as follows :

X-IBM-Client-Id	String containing the client ID of the application invoking IRAS API. This value will be provided to the application vendor by IRAS. E.g. a1234b5c-1234-abcd-efgh-a1234b5cdef
X-IBM-Client-Secret	String containing the client secret of the application invoking IRAS API. This value will be provided to the application vendor by IRAS. E.g. a12345bC67e8fG9a12345bC67e8fG9a12345bC67e8fG9

This document serves as a guide for developers to consume IRAS API services.

2. Registration

Developers are required to test APIs in the Sandbox environment first, before using them in Production. Create a developer account at https://apisandbox.iras.gov.sg/ to subscribe to IRAS APIs for Sandbox Testing, and a developer account at https://apiservices.iras.gov.sg/ to subscribe to IRAS APIs for Production use.

Note: In order for IRAS to identify your API subscriptions, please follow the steps below to set up your account:

Step 1: Create Account

- Create an account at the IRAS API Marketplace with your personal particulars. A computer-generated email will be sent to you for account activation.

- After account activation and login, click on your Username at the top-right corner of the page > My Organization > Edit organization.

Note: Organization name should reflect your actual company / organisation name (followed by the ROC or UEN where applicable), e.g. ABC Pte Ltd_201712345X.

*   *Take note of the underscore after the company / organisation name.*

Step 2: Register App

- Click on Apps on the Menu Bar > Register New Application

- App Title will be the name displayed to your clients (i.e. software users) upon request to give consent for your App to call IRAS' API.

Step 3: Subscribe

- Click on API Products > Select the Submission of Donation Records (Corppass) API Product > Click Subscribe

- Select your registered App > Click Subscribe

Refer to https://go.gov.sg/iras-apiuserguide for more details on the steps above.

3. API Services

Name of API Service	Description	Endpoint
Submission of Tax-Deductible Donation information by IPCs/ Grantmakers	This Corppass authenticated API allows software developers and IPCs/Grantmakers to incorporate within the software, the feature for approved IPCs/Grantmakers to validate and submit tax-deductible donation information to IRAS directly. The submitted information will be auto included into donors' tax assessment.	For Sandbox Testing: https://apisandbox.iras.gov.sg/iras/sb/DonationCP/submit For Production Usage: https://apiservices.iras.gov.sg/iras/prod/DonationCP/submit

To consume this API service, you must be subscribed to Corppass Authentication API. Refer to IRAS Corppass Authentication API Services Interface Specification for more information.

3.1 Network Security

The API endpoint has to be triggered from a Server-to-Server connection. Transport Layer Security (TLS) and IP Whitelisting will be used to secure end-to-end communications between the server invoking the API and IRAS API Gateway.

Your server, software or application must be able to:

1. Support the following protocols: HTTP/2, TLS 1.2/1.3, and

2. Trigger HTTP/GET and HTTP/POST requests.

3.2 Message Security

The following parameters must be populated in the HTTP header:

X-IBM-Client-Id	String containing the client ID of the application invoking IRAS API. This value will be provided to the application vendor by IRAS.
X-IBM-Client-Secret	String containing the client secret of the application invoking IRAS API. This value will be provided to the application vendor by IRAS.
access_token	JWT (JSON Web Token) Authorisation Token Can be blank if validateOnly parameter is set to 'true'. Refer to 4.2.

	Refer to IRAS CorpPass Authentication API Services Interface Specification for more information on obtaining the access token. The required access scope for this API is *DonationSub*.
Content-Type	application/json

Sample HTTP Header in a HTTP Post Request

```
POST /iras/prod/DonationCP/submit HTTP/1.1
Host:  https://apiservices.iras.gov.sg
X-IBM-Client-Id:  40e7be2f-0b4f-4985-bcc9-cdfd38c5b5c8
X-IBM-Client-Secret: vD0kR8iT3kR1hB8dP1qS3hC4lJ1aA1fV4pQ0uW0hI3uI5bW4rU
access_token:  eyAidHlwIjogIkpXVCIsICJhbGciOiAiUlMyNTYiIH0…….
Content-Type:  application/json
Accept:   application/json
```

3.3   General Usage

Each API request payload is limited 5,000 records or 2 MB in total data size, whichever is lower.

3.4   Sandbox Usage

Developers must be subscribed to the "Submission of Donation Records (Corppass) (Sandbox) API before commencing sandbox testing. This service in the sandbox environment is designed to mimic the production environment so that developers can test the API integration before submitting actual data to the production environment. IRAS' approval is not required for subscription to this service. The Sandbox test scenarios and entities will be published on the IRAS website.

After a successful sandbox submission, an acknowledgement code will be generated and provided within the API response.

Submit the acknowledgement code(s) for each of the test scenarios to data_mgmt@iras.gov.sg for verification. Refer to the Donation API Test Guide.

Developers who have passed sandbox testing can proceed to subscribe to the "Submission of Donation Records (Corppass)" API in the API Portal for Production.

Note: You must subscribed to Corppass Authentication (Sandbox) API before you can commence sandbox testing.

3.5   Production Usage

The "Submission of Donation Records (Corppass)" API in Production requires approval by IRAS, subject to successful validation of sandbox testing. Please ensure that a subscription request is also made to the "Corppass Authentication" API for approval to be granted.

After approval is granted by IRAS, client application can POST a JSON request object to the production URL.

Use of IRAS API(s) are governed by the Additional API Terms of Use found at the API Marketplace.

The use of IRAS API(s) in the Production environment is only intended for the purpose as indicated in the API service description. Any other forms of use, **including all forms of tests**, are strictly prohibited. In the event such activities are found to have occurred in the Production environment, you will be deemed to have breached the API Terms of Use, which may result in the suspension or termination of your account.

3.6   Common Interface Information

- JSON is case sensitive by specifications.

- All date strings are to be represented in compliance to the ISO-8601 standard.

- All string fields are subject to validation of the following acceptable characters that is allowed (in red):

    o [A-Za-z0-9&'()+,-./:;=?@_`{}~!#%\<>|[ ]*$] (**Note**: whitespace is included)

- There is no partial processing of any request payloads. Submissions will only be successfully processed when there are no validation errors in all records.

- All properties follow the camel-case convention.

- Unless stated as optional, all JSON object properties must be specified.

- Unless otherwise specified, all JSON services are invoked using HTTP verb POST.

- All input data format are as specified like the following:

Data Type and Size	Description	Example
String(12)	A string containing maximum 12 characters.	"180084010K"
Number(4)	A numeric value containing maximum 4 whole numbers.	1990

3.7   Common Response Payload

All response payloads share the following common fields:

Parameter Name	Data Type	Description
data	Object	The data property will be populated differently based on the API that is being invoked.
returnCode	Integer	**10** : Success - The request was successfully processed. **30** : Failure – The request was not processed. Refer to "info" object for error information.
info	Object	This complex object holds any diagnostic information that will allow developers to debug their failed requests.
info.message	String	Diagnostic message in the event of warning or error.
Info.messageCode	Integer	Integer code signifying the type of error or warning. **850300 :** Request object is null – The incoming JSON request is null. **850301** : Arguments error – There is an error with one of the arguments provided. **850302** : Generic error – There is an exception within the service. **850303** : Service is inactive. **850304** : Service is not authorized for usage based on the provided credentials. **850305** : Invalid test user – The input fields provided are not valid for sandbox testing.
info.fieldInfoList	Array	An array for FieldInfo objects.
info.fieldInfoList.field	String	Name of the field that resulted in a warning / error.
Info.fieldInfoList.message	String	Diagnostic message provided to aid consumer's developers.
Info.fieldInfoList.recordID	String	The unique identification number of the affected record. Record IDs with similar error will be lumped together. E.g. "recordID": "3,5,7" Note: This is only present if the field resulting in an error belongs to a donationDtl object.

4. Submission of Donation Records (Corppass) API Service

4.1 API Endpoint

Method	Endpoint
POST	/DonationCP/submit

4.2 Request Payload

Parameter Name	Data Type (Max Length)	Mandatory / Optional / Conditional	Description
orgAndSubmissionInfo	Object	M	This complex object holds the IPC details and information of donation submission.
validateOnly	Boolean	M	Must be 'true' or 'false' only. If validateOnly = true, the API will perform validation of the donation information without submission. If validateOnly = false, the API will perform validation of the donation information and submission to IRAS.
basisYear	Number (4)	M	The year for which donation is to be reported. Year of Information is to be input as '2021' for donation made in year 2021. YearInfo can cater to 3 back years, current year and 1 advance year.
organisationIDType	String (1)	M	Must be 5,6,U,A or I. 5= UEN-BUSINESS 6= UEN-LOCAL CO U= UEN-OTHERS A= ASGD I= ITR
organisationIDNo	String (12)	M	Must be one of the following: UEN-BUSINESS = NNNNNNNNC UEN-LOCAL CO = YYYYNNNNNC UEN-OTHERS = TYYPQNNNNC or SYYPQNNNNC ASGD No. = ANNNNNNNC ITR No. = NNNNNNNNNC Where, C = Check digit N = Numeric Y = Year
organisationName	String (60)	M	Organisation Name

batchIndicator	String (1)	M	Must be O or A O = Original A = Amendment An amendment submission will overwrite the previous donation records submitted. Example: <u>Original Submission:</u> NRIC: SNNNNNNNA Date of donation: 15/01/2023 Receipt No: A10001 Donation Amount: $100 <u>Amendment Submission:</u> NRIC: Same as in original Date of donation: 25/01/2023 Receipt No: Same as in original Donation Amount: $50 The date of donation and donation amount will be overwritten in IRAS' records with the latest information based on the amendment submission.
authorisedPersonIDNo	String (12)	M	The ID Number of person submitting the Donation Information. Must be one of the following: Valid NRIC with prefix S/T Valid FIN with prefix F/G/M Valid ASGD as ANNNNNNNC Valid ITR as NNNNNNNNNC
authorisedPersonName	String (30)	M	The name of the person submitting the Donation information.
authorisedPersonDesignation	String (30)	O	The designation of the person submitting the Donation information
telephone	String (20)	M	Telephone number of the person submitting the Donation information. Must be at least 8 digits.
authorisedPersonEmail	String (50)	M	Email address of the person submitting the Donation information.
numOfRecords	Number (7)	M	Total number of Donation earners in the *donationDtl*. Accepting Integers only.
totalDonationAmount	Number (12)	M	Sum of donation amount in *donationDtl*.

donationDonorDtl	Object	M	This complex object holds the donation entered details
recordID	Number (4)	M	The running number assigned should start from 1 and be unique.
idType	String (1)	M	Must be one of the following: 1 = NRIC 2 = FIN 5 = UEN-BUSINESS 6 = UEN-LOCAL CO A = ASGD I = ITR U = UEN-OTHERS
idNumber	String (12)	M	Must be one of the following: Valid NRIC with prefix S/T Valid FIN with prefix F/G/M Valid UEN-BUSINESS as NNNNNNNNC Valid UEN-LOCAL CO as YYYYNNNNNC Valid ASGD as ANNNNNNNC Valid ITR as NNNNNNNNNC Valid UEN-OTHERS as TyyPQnnnnC or SyyPQnnnnC
individualIndicator	String (3)	O	Individual (IND) or Non-Individual Indicator (NON). Mandatory, if ID Type=A(AGSD) or I=ITR To pass either IND or NON.
name	String (80)	M	The donor's name
addressLine1	String (30)	O	The donor's address line 1
addressLine2	String (30)	O	The donor's address line 2
postalCode	Number (6)	O	Must be numeric
donationAmount	Number (9)	M	Must not be blank and must not have space(s). Exclude decimal. Always round up the donation amount if the total has a decimal e.g. If the donation amount = $101.20, then output as =102. Must not be $0 for Original records i.e. if batchIndicator=O
dateOfDonation	Number (8)	M	Must be in the format of YYYYMMDD
receiptNum	String (10)	M	Receipt number or Reference number unique to the donation record

typeOfDonation	String (1)	M	Must be one of the following values: O = Outright Cash S = Share/Unit Trust A = Artifact L = Land/Building P = Public Sculpture If TypeofDonation=S, the Donor ID Type must be NRIC or FIN or ASGD or ITR. If ASGD or ITR, the individualIndicator must be =IND
namingDonation	String (1)	M	Must be one of the following values: Z = Non naming donation (by default) N = Name IPC (where the IPC (e.g. foundation) is named after the donor or his immediate and living family member. F = Name Facility (where a facility belonging to an IPC has been named after the donor.)

```
{
  "orgAndSubmissionInfo": {
    "validateOnly": "false",
    "basisYear": "2020",
    "organisationIDType": "5",
    "organisationIDNo": "199900001M",
    "organisationName": "ABCD Medical Institute",
    "batchIndicator": "O",
    "authorisedPersonIDNo": "S12345678A",
    "authorisedPersonName": "James Tan",
    "authorisedPersonDesignation": "Manager",
    "telephone": "61234567",
    "authorisedPersonEmail": "james.tan@sunny.com.sg",
    "numOfRecords": "2",
    "totalDonationAmount": "5000",
  },
  "donationDonorDtl": [
    {
      "recordID": "1",
      "idType": "1",
      "idNumber": "S1234567Z",
      "individualIndicator": "",
      "name": "James Zhang",
      "addressLine1": "",
      "addressLine2": "",
      "postalCode": "640501",
      "donationAmount": "2500",
      "dateOfDonation": "20200105",
      "receiptNum": "RECP123",
      "typeOfDonation": "O",
      "namingDonation": "Z"
    },
    {
      "recordID": "2",
      "idType": "A",
      "idNumber": "S12345678A",
      "individualIndicator": "IND",
      "name": "Andrew Chan",
      "addressLine1": "",
      "addressLine2": "",
      "postalCode": "640501",
      "donationAmount": "2500",
      "dateOfDonation": "20200505",
      "receiptNum": "RECP128",
```

4.3   Response Payload

Parameter Name	Data Type	Description
data	Object	The object payload containing the information for the queried document reference number and stamp certificate reference.
data.acknowledgementCode	Number	System generated acknowledgement code
returnCode	As per [section 3.7](#)	
info		
info.message		
info.messageCode		
info.fieldInfoList		
info.fieldInfoList.field		
info.fieldInfoList.message		
info.fieldInfoList.recordID		

Sample success JSON response payload, when validateOnly=false

```
{
  "returnCode": "10",
  "data": {
    "acknowledgementCode": "ES20070621DONS12486243",
  },
  "info": {
    "fieldInfoList": []
  }
}
```

Sample success JSON response payload, when validateOnly=false

```
{
  "returnCode": "30",
  "info": {
    "messageCode": "850301",
    "message": "Arguments Error",
    "fieldInfoList": [
      {
        "field": "idNumber",
         "message": "Incorrect ID Type and/or ID No",
        "recordID": "3,5,7"
      },
      {
        "field": "name",
         "message": "Must not be blank",
        "recordID": "3,5,7"
      }
    ]
  }
}
```

For Validate only API calls

Sample success JSON response payload, when validateOnly=true

```
{
  "returnCode": "10",
  "data": {},
  "info": {
    "fieldInfoList": []
  }
}
```

Sample success JSON response payload, when validateOnly=true

```
{
  "returnCode": "30",
  "info": {
    "messageCode": "850301",
    "message": "Arguments Error",
    "fieldInfoList": [
      {
        "field": "idNumber",
         "message": "Incorrect ID Type and/or ID No",
        "recordID": "3,5,7"
      }
    ]
  }
}
```

5. Sample Code (C#)

```csharp
using System;
using System.Net;
using System.IO;
using System.Text;

// jsonData – contains data from Section 3.1.1 of this document
public static void callWebAPI(string jsonData, string url)
{
        try
        {
            var httpWebRequest = (HttpWebRequest)WebRequest.Create(url);
            httpWebRequest.ContentType = "application/json;";
            httpWebRequest.Method = "POST";

            //Step 1: Enter the Client-Id given by IRAS
            httpWebRequest.Headers["X-IBM-Client-Id"] = "{YOUR_CLIENT_ID}";
            //Step 2: Enter the Client-Secret given by IRAS
            httpWebRequest.Headers["X-IBM-Client-Secret"] ="{YOUR_CLIENT_SECRET}";

            // Step 3: Call API using POST
            using (var streamWriter = new StreamWriter(httpWebRequest.GetRequestStream()))
            {
                streamWriter.Write(jsonData);
                streamWriter.Flush();
                streamWriter.Close();
            }

            // Step 3a: Output response
            var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse();
            using (var streamReader = new StreamReader(httpResponse.GetResponseStream()))
            {
                var result = streamReader.ReadToEnd();
                //print the received reponse
                Console.WriteLine(result);
            }
        }
        catch (WebException e)
        {
            if (!string.IsNullOrEmpty(e.Message))
            {
                // Step 3b: Print general errors
                Console.WriteLine("Exception - ");
                Console.WriteLine(e.Message);
            }

            if (e.Response != null)
            {
                // Step 3c: Print Output response exception
                Stream receiveStream = e.Response.GetResponseStream();
                StreamReader readStream = new StreamReader(receiveStream, Encoding.UTF8);
                // print the error received from Server
                Console.WriteLine("Response error received - ");
                Console.WriteLine(readStream.ReadToEnd());
            }
        }
}
```