IRAS API SERVICES INTERFACE SPECIFICATIONS

Corporate Rental Submission

Last updated on: 7 June 2022

Version No.: 1.04

Version Control

Version History

Revision Date	Version Number	Change Log Summary
07 Jun 2022	1.04	Revised copy

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 easy integration between IRAS' service with external developers.

There will be a variety of services available in due time. 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 to help developers consume the API services provided by IRAS.

2. Registration at API Portal

Application developers are required

- To create a developer's account at https://apisandbox.iras.gov.sg/ and subscribe to IRAS API services for Sandbox Testing, and
- To create a developer's account at https://apiservices.iras.gov.sg/ and subscribe to IRAS API services for Production usage.

Note: In order for IRAS to identify your API subscriptions, please enter your details as follows when creating an account:

- "First name" field: To enter "Name of organisation"
- "Last name" field: To enter "Tax Reference No. of organisation"

A computer-generated email will be sent to the subscriber's email account for account activation of the API Portal.

3. API Services

The following sections describe the request and the response of the API service.

S/No	Name of API Services	Description	URL
1	Corporate Rental Submission	This API can be used to submit rental information for property tax assesments.	For Sandbox Testing: https://apisandbox.iras.gov.sg/iras/sb/rental/submission
			For Production Usage: https://apiservices.iras.gov.sg/iras/prod/rental/submission

3.1 General Information

Each submission of the rental information cannot be more than 2 MB in data size.

3.1.1 Sandbox Usage

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.

After a successful sandbox submission, a reference number will be generated and provided within the API response.

Submit the reference number to Tricia_ANG@iras.gov.sg and Desyanti_TAN@iras.gov.sg for registration and subscription to the IRAS API services for Production Usage.

3.1.2 Production Usage

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

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.

3.1.3 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)
    - [a-zA-Z0-9‘@#()\-,./&+_] (**Note**: whitespace is included)

- 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 Format Specification

Data Type and Size	Description	Example
String(12)	A string containing maximum 12 characters.	"S1234567Z"
Number(4)	A numeric value containing maximum 4 whole numbers.	1990

3.1.4 Common Response Payload

All response payloads share the following common fields:

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. **20** : Warning - The request was successfully processed. However, there are non-fatal issues. Please refer to the "info" object for diagnostic information. **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.

3.2 Corporate Rental Submission

3.2.1 Request Payload

Fields	Type	Mandatory	Descriptions
orgAndSubmissionInfo	Object	Y	This complex object holds organisation details and information of rental submission.
developmentName	String(60)	Y	The development name.
assmtYear	Number(4)	Y	General assessment year
authorisedPersonName	String(30)	Y	The name of the person submitting the Rental information.
authorisedPersonEmail	String(50)	Y	Email address of the person submitting the Rental information.
propertyDtl	Array	Y	This complex object holds the property informations.
recordID	String(10)	Y	The running number assigned should start from 1 and be unique.
propertyTaxRef	String(8)	Y	Property Tax Reference number. Compulsory except for newly configured unit.
unitNo	String(20)	Y	Unit No. Compulsory.
letArea	String(20)	Y	Let area in m2 (in 2 decimal point). Compulsory.
vacantInd	String(1)	Y	Indicates whether the unit is a vacant unit. To indicate 'Y' if the unit is vacant. Default is 'N'.
tenantName	String(100)	Y	Tenant name
netRentAmt	Number(20)	Y	Monthly net rental amount (in 2 decimal point). Compulsory except for vacant unit.
svcChargeAmt	Number(20)	N	Monthly service charge amount (in 2 decimal point)..
advPromotionAmt	Number(20)	N	Monthly advertising and promotion amount (in 2 decimal point).
dateLeaseStart	String(8)	Y	Lease start date, in the format of YYYYMMDD. Compulsory except for vacant unit.
dateLeaseEnd	String(8)	Y	Lease end date, in the format of YYYYMMDD. Compulsory except for vacant unit.
GTOAmt	Number(20)	N	Annual gross turnover rent amount (in 2 decimal point).
GTOInfo	String(100)	N	GTO Structure information.
dateGTOStart	Number(8)	N	GTO start date, in the format of YYYYMMDD.
dateGTOEnd	Number(8)	N	GTO end date, in the format of YYYYMMDD.
infoRemarks	String(150)	N	Remarks

Sample JSON request payload
(*Note: one propertydtl record)


```
{
  "orgAndSubmissionInfo": {
    "developmentName": "BugisCentre",
    "assmtYear": 2018,
    "authorisedPersonName": "Steve",
    "authorisedPersonEmail": "Steve@bugiscentre.com.sg"
    },
    "propertyDtl": [
      { "recordID": "1",
        "propertyTaxRef": "0200320A",
        "unitNo": "#01-01",
        "letArea": 80.00,
        "vacantInd": "N",
        "tenantName": "BURGER KING SINGAPORE PTE. LTD",
        "netRentAmt": 6888.96,
        "svcChargeAmt": 1722.24,
        "advPromotionAmt": 516.67,
        "dateLeaseStart": "20170101",
        "dateLeaseEnd": "20180106",
        "GTOAmt": 149575.67,
        "GTOInfo": "Rental + 0.5 of GTO OR 15 of GTO whichever
is higher",
        "dateGTOStart": 20170101,
        "dateGTOEnd": 20171231,
        "infoRemarks": "any other remarks"}
]}
```

Sample JSON request payload
(*Note: more than one propertydtl records)

```
{
  "orgAndSubmissionInfo": {
    "developmentName": "BugisCentre",
    "assmtYear": 2018,
    "authorisedPersonName": "Steve",
    "authorisedPersonEmail": "Steve@bugiscentre.com.sg"
    },
    "propertyDtl": [
      { "recordID": "1",
        "propertyTaxRef": "0200320A",
        "unitNo": "#01-01",
        "letArea": 80.00,
        "vacantInd": "N",
        "tenantName": "BURGER KING SINGAPORE PTE. LTD",
        "netRentAmt": 6888.96,
        "svcChargeAmt": 1722.24,
        "advPromotionAmt": 516.67,
        "dateLeaseStart": "20170101",
        "dateLeaseEnd": "20180106",
        "GTOAmt": 149575.67,
        "GTOInfo": "Rental + 0.5 of GTO OR 15 of GTO whichever
is higher",
        "dateGTOStart": 20170101,
        "dateGTOEnd": 20171231,
        "infoRemarks": "any other remarks"},
      { "recordID": "2",
        "propertyTaxRef": "0200323U",
        "unitNo": "01-02",
        "letArea": 100.20,
        "vacantInd": "Y",
        "tenantName": "",
        "GTOInfo": "",
        "infoRemarks": ""}
]}
```

3.2.2 Response Payload

data	Object	The object payload containing the reference code of successful submission.
data.refNo	String(30)	Reference number of successful submission.
returnCode	As per Section 3.1.4	
info		
info.message		
info.messageCode		
info.fieldInfoList		
info.fieldInfoList.field		
info.fieldInfoList.message		

info.fieldInfoList.recordID	String(10)	Record ID

Sample success JSON response payload
(*Note: refNo may not be the same per below.)

```
{
  "returnCode": 10,
  "data": {
    "refNo": "PTSTMT1113525"
  },
  "info": {
    "fieldInfoList": []
  }
}
```

Sample error JSON response payload

```
{
  "returnCode": 30,
  "info": {
    "messageCode": 850301,
    "message": "Arguments Error",
    "fieldInfoList": [
      {
        "field": "propertyTaxRef",
        "message": "Invalid ID ",
        "recordID": "1"
      },
      {
        "field": "letArea",
        "message": "Must not be blank",
        "recordID": "2"
      }
    ]
  }
}
```

3.2.3 Sample Scenarios

Below are some examples on how to populate the input parameters based on some scenarios provided here.

4. Sample Code (C#)

```
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["Client-Id"] = "{YOUR_CLIENT_ID}";
            //Step 2: Enter the Client-Secret given by IRAS
            httpWebRequest.Headers["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());
            }
        }
}
```

The information provided is intended for better general understanding and is not intended to comprehensively address all possible issues that may arise. The contents are correct as at 24 Jan 2018 and are provided on an "as is" basis without warranties of any kind. IRAS shall not be liable for any damages, expenses, costs or loss of any kind however caused as a result of, or in connection with your use of this document.

While every effort has been made to ensure that the above information is consistent with existing policies and practice, should there be any changes, IRAS reserves the right to vary our position accordingly.