

腾讯云微搭低代码

内置 API 专区



腾讯云

【 版权声明 】

©2013–2026 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【 商标声明 】



及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【 服务声明 】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【 联系我们 】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或 95716。

文档目录

内置 API 专区

前端 API 概览

服务端 API

概述

内置 API 专区

前端 API 概览

最近更新时间：2024-11-14 10:35:12

通过微搭的前端 API，您可以拿到当前应用的数据和常见的工具库，可以调用云函数，也可以直接在低码编辑器中直接使用 API 等，增加编写程序的扩展性、便利性。

界面交互相关接口

接口名称	接口功能
showToast	显示提示框
showLoading	显示 loading 提示框
hideLoading	隐藏 loading 提示框
showModal	显示模态对话框

更多交互类相关前端 API 请参见 [微搭 API > 工具交互方法](#)。

数据源、流程以及其他云端能力接口

接口名称	接口功能
\$w.cloud.callDataSource	获取数据源信息
\$w.cloud.callWorkflow	调用微搭流程服务

更多其他云端能力接口，请参见 [微搭 API > 数据源/流程/云端能力](#)。

更多平台相关前端 API 接口

了解更多微搭前端 API 请参见 [微搭前端 API 列表](#)。

服务端 API

概述

最近更新时间：2024-09-30 14:33:41

腾讯云微搭低代码通过开放接口来满足第三方服务的定制化开发需求。

开放功能

产品功能	说明	接口文档
数据模型	通过开放接口对数据模型进行新增、删除、更新、查询操作	查看
用户权限	通过开放接口对用户/部门/角色/权限进行管理	查看
工作流	通过开放接口对工作流的流程实例、任务进行管理	查看

接口使用

域名

```
https://<环境ID>.ap-shanghai.tcb-api.tencentcloudapi.com
```

ⓘ 说明：

环境 ID 可前往 [资源管理](#) 页面获取。

操作步骤

- 前往 [腾讯云控制台](#) 申请 SecretId + SecretKey。
- 使用 OAuth 2.0 鉴权方式换取 Access Token。
- 使用 Access Token 请求开放接口并在 Request Header 中加入 `Authorization: Bearer <Access Token>`。

请求说明

- API 的所有接口均通过 HTTPS 进行通信，均使用 UTF-8 编码。
- 支持的 HTTP 请求方法：POST、GET、PUT、PATCH、DELETE。
- Content-Type 类型：`application/json;utf-8`。

参数说明

接口包含三种参数类型：

- **url**: 位于请求路由，形如 `GET /weda/odata/v1/pre/data_xxxx`。
- **queryString**: 位于 `?` 后，形如 `GET /weda/odata/v1/pre/data_xxxx?$filter=name eq '张三'`。
- **body**: 位于 POST 请求体，以标准 `json` 传入，形如 `POST /weda/odata/v1/pre/data_xxxx { "name": "张三" }`。

代码示例

NodeJS

```
const Koa = require('koa');
const fetch = require('node-fetch');
const app = new Koa();

const EnvId = ''; // 环境 ID, 例如 lowcode-2gay8jgh25
const SecretId = '';
const SecretKey = '';

// 域名
const domain = `https://${EnvId}.ap-shanghai.tcb-api.tencentcloudapi.com`;

app.use(async ctx => {
  // 换取 AccessToken
  const tokenResponse = await
  fetch(`${domain}/auth/v1/token/clientCredential`, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": `Basic
      ${Buffer.from(`${SecretId}:${SecretKey}`).toString('base64')}`
    },
    body: JSON.stringify({
      grant_type: 'client_credentials',
    })
  });
});
```

```
const { access_token } = await tokenResponse.json();

// 请求某个服务端 API
const queryResponse = await
fetch(`${domain}/weda/odata/v1/prod/sys_user`, {
  method: "GET",
  headers: {
    "Content-Type": "application/json",
    "Authorization": `Bearer ${access_token}`
  }
});

ctx.body = await queryResponse.json();
});

app.listen(3000);
```

Java

```
package com.example.odata;

import org.apache.http.HttpEntity;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.ResponseHandler;
import org.apache.http.client.methods.HttpDelete;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.util.EntityUtils;
import org.springframework.util.Base64Utils;

import com.fasterxml.jackson.databind.ObjectMapper;

import java.util.HashMap;
import java.util.Map;
```

```
public class OpenApiClient {
    // 获取 token
    private String getToken(String envId, String secretId, String
secretKey) {
        String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
        String url = host + "/auth/v1/token/clientCredential";
        HttpPost httpPost = new HttpPost(url);
        String basicKey = secretId + ":" + secretKey;
        String authorizationKey = "Basic " +
Base64Utils.encodeToString(basicKey.getBytes());
        httpPost.addHeader("Authorization", authorizationKey);
        httpPost.addHeader("Content-Type", "application/json");
        Map<String, String> body = new HashMap<>();
        body.put("grant_type", "client_credentials");
        ObjectMapper mapper = new ObjectMapper();
        try {
            String bodyStr = mapper.writeValueAsString(body);
            StringEntity requestBody = new StringEntity(bodyStr,
"UTF-8");
            httpPost.setEntity(requestBody);
        } catch (Exception e) {
            System.out.println(e.toString());
            return "";
        }
        ResponseHandler<String> responseHandler = response -> {
            int status = response.getStatusLine().getStatusCode();
            if (status >= 200 && status < 300) {
                HttpEntity entity = response.getEntity();
                return entity != null ? EntityUtils.toString(entity)
: null;
            } else {
                throw new ClientProtocolException("Unexpected
response status: " + status +
EntityUtils.toString(response.getEntity()));
            }
        };
        try {
```

```
        CloseableHttpClient httpClient =
HttpClients.createDefault();
        String responseBody = httpClient.execute(httpPost,
responseHandler);
        Map<String, Object> responseMap =
mapper.readValue(responseBody, Map.class);
        /*
        {
            "token_type": "Bearer",
            "access_token": "",
            "expires_in": 111, // 过期时间,单位为 s
        }
        */
        return "Bearer " +
responseMap.get("access_token").toString();
    } catch (Exception e) {
        System.out.println(e.toString());
    }
    return "";
}

// GET: 获取数据源记录列表
private void GET(String token, String envId, String envType,
String datasourceName) {
    String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
    String url = host + "/weda/odata/v1/" + envType + "/" +
datasourceName;
    HttpGet httpGet = new HttpGet(url);
    httpGet.addHeader("Authorization", token);
    httpGet.addHeader("Content-Type", "application/json");
    ResponseHandler<String> responseHandler = response -> {
        int status = response.getStatusLine().getStatusCode();
        if (status >= 200 && status < 300) {
            HttpEntity entity = response.getEntity();
            return entity != null ? EntityUtils.toString(entity)
: null;
        } else {
            throw new ClientProtocolException("Unexpected
response status: " + status +
```

```
EntityUtils.toString(response.getEntity()));
    }
};
try {
    CloseableHttpClient httpClient =
HttpClients.createDefault();
    String responseBody = httpClient.execute(httpGet,
responseHandler);
    ObjectMapper mapper = new ObjectMapper();
    Map<String, Object> responseMap =
mapper.readValue(responseBody, Map.class);
    // 打印 body
    System.out.println(responseMap.toString());
} catch (Exception e) {
    System.out.println(e.toString());
}
}

// GET: 获取数据源记录详情
private void GET(String token, String envId, String envType,
String datasourceName, String recordId) {
    String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
    String url = host + "/weda/odata/v1/" + envType + "/" +
datasourceName + "(" + recordId + ")";
    HttpGet httpGet = new HttpGet(url);
    httpGet.addHeader("Authorization", token);
    httpGet.addHeader("Content-Type", "application/json");
    ResponseHandler<String> responseHandler = response -> {
        int status = response.getStatusLine().getStatusCode();
        if (status >= 200 && status < 300) {
            HttpEntity entity = response.getEntity();
            return entity != null ? EntityUtils.toString(entity)
: null;
        } else {
            throw new ClientProtocolException("Unexpected
response status: " + status +
EntityUtils.toString(response.getEntity()));
        }
    };
};
```

```
try {
    CloseableHttpClient httpClient =
HttpClients.createDefault();
    String responseBody = httpClient.execute(httpGet,
responseHandler);
    ObjectMapper mapper = new ObjectMapper();
    Map<String, Object> responseMap =
mapper.readValue(responseBody, Map.class);
    // 打印 body
    System.out.println(responseMap.toString());
} catch (Exception e) {
    System.out.println(e.toString());
}
}

// POST: 创建记录
private void POST(String token, String envId, String envType,
String datasourceName, Map<String, Object> jsonBody) {
    String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
    String url = host + "/weda/odata/v1/" + envType + "/" +
datasourceName;
    HttpPost httpPost = new HttpPost(url);
    ObjectMapper mapper = new ObjectMapper();
    try {
        String bodyStr = mapper.writeValueAsString(jsonBody);
        StringEntity requestBody = new StringEntity(bodyStr,
"UTF-8");
        httpPost.setEntity(requestBody);
    } catch (Exception e) {
        System.out.println(e.toString());
        return;
    }
    httpPost.addHeader("Authorization", token);
    httpPost.addHeader("Content-Type", "application/json");
    ResponseHandler<String> responseHandler = response -> {
        int status = response.getStatusLine().getStatusCode();
        if (status >= 200 && status < 300) {
            HttpEntity entity = response.getEntity();
```

```
        return entity != null ? EntityUtils.toString(entity)
: null;
    } else {
        throw new ClientProtocolException("Unexpected
response status: " + status +
EntityUtils.toString(response.getEntity()));
    }
};
try {
    CloseableHttpClient httpClient =
HttpClient.createDefault();
    String responseBody = httpClient.execute(httpPost,
responseHandler);
    Map<String, Object> responseMap =
mapper.readValue(responseBody, Map.class);
    // 打印 body

System.out.println("aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa" +
responseMap.toString());
    } catch (Exception e) {
        System.out.println(e.toString());
    }
}

// PUT: 更新记录
private void PUT(String token, String envId, String envType,
String datasourceName, String recordId, Map<String, Object>
jsonBody) {
    String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
    String url = host + "/weda/odata/v1/" + envType + "/" +
datasourceName + "(" + recordId + ")";
    HttpPost httpPost = new HttpPost(url);
    ObjectMapper mapper = new ObjectMapper();
    try {
        String bodyStr = mapper.writeValueAsString(jsonBody);
        StringEntity requestBody = new StringEntity(bodyStr,
"UTF-8");
        httpPost.setEntity(requestBody);
    } catch (Exception e) {
```

```
        System.out.println(e.toString());
        return;
    }
    httpPost.addHeader("Authorization", token);
    httpPost.addHeader("Content-Type", "application/json");
    ResponseHandler<String> responseHandler = response -> {
        int status = response.getStatusLine().getStatusCode();
        if (status >= 200 && status < 300) {
            HttpEntity entity = response.getEntity();
            return entity != null ? EntityUtils.toString(entity)
: null;
        } else {
            throw new ClientProtocolException("Unexpected
response status: " + status +
EntityUtils.toString(response.getEntity()));
        }
    };
    try {
        CloseableHttpClient httpClient =
HttpClient.createDefault();
        String responseBody = httpClient.execute(httpPost,
responseHandler);
        Map<String, Object> responseMap =
mapper.readValue(responseBody, Map.class);
        // 打印 body
        System.out.println(responseMap.toString());
    } catch (Exception e) {
        System.out.println(e.toString());
    }
}

// DELETE: 删除记录
private void DELETE(String token, String envId, String envType,
String datasourceName, String recordId) {
    String host = "https://" + envId + ".ap-shanghai.tcb-
api.tencentcloudapi.com";
    String url = host + "/weda/odata/v1/" + envType + "/" +
datasourceName + "(" + recordId + ")";
    HttpDelete httpDelete = new HttpDelete(url);
    httpDelete.addHeader("Authorization", token);
}
```

```
httpDelete.setHeader("Content-Type", "application/json");
ResponseHandler<String> responseHandler = response -> {
    int status = response.getStatusLine().getStatusCode();
    if (status >= 200 && status < 300) {
        HttpEntity entity = response.getEntity();
        return entity != null ? EntityUtils.toString(entity)
: null;
    } else {
        throw new ClientProtocolException("Unexpected
response status: " + status +
EntityUtils.toString(response.getEntity()));
    }
};
try {
    CloseableHttpClient httpClient =
HttpClient.createDefault();
    httpClient.execute(httpDelete, responseHandler);
} catch (Exception e) {
    System.out.println(e.toString());
}
}

public static void main(String[] args) {
    OpenApiClient openApiClient = new OpenApiClient();
    // 真实环境 ID
    String envId = "";
    String secretId = "";
    String secretKey = "";
    // 数据模型标识
    String datasourceName = "";
    // 数据模型类型
    String envType = "pre";
    String token = openApiClient.getToken(envId, secretId,
secretKey);

    // 创建记录
    Map<String, Object> postBody = new HashMap<>();
    postBody.put("name", "zhangsan");
    postBody.put("age", 12);
```

```
/*
// 创建记录
openApiClient.POST(token, envId, envType, datasourceName,
postBody);

//删除记录
openApiClient.DELETE(token, envId, envType, datasourceName,
"");

//查看
openApiClient.GET(token, envId, envType, datasourceName,
"");

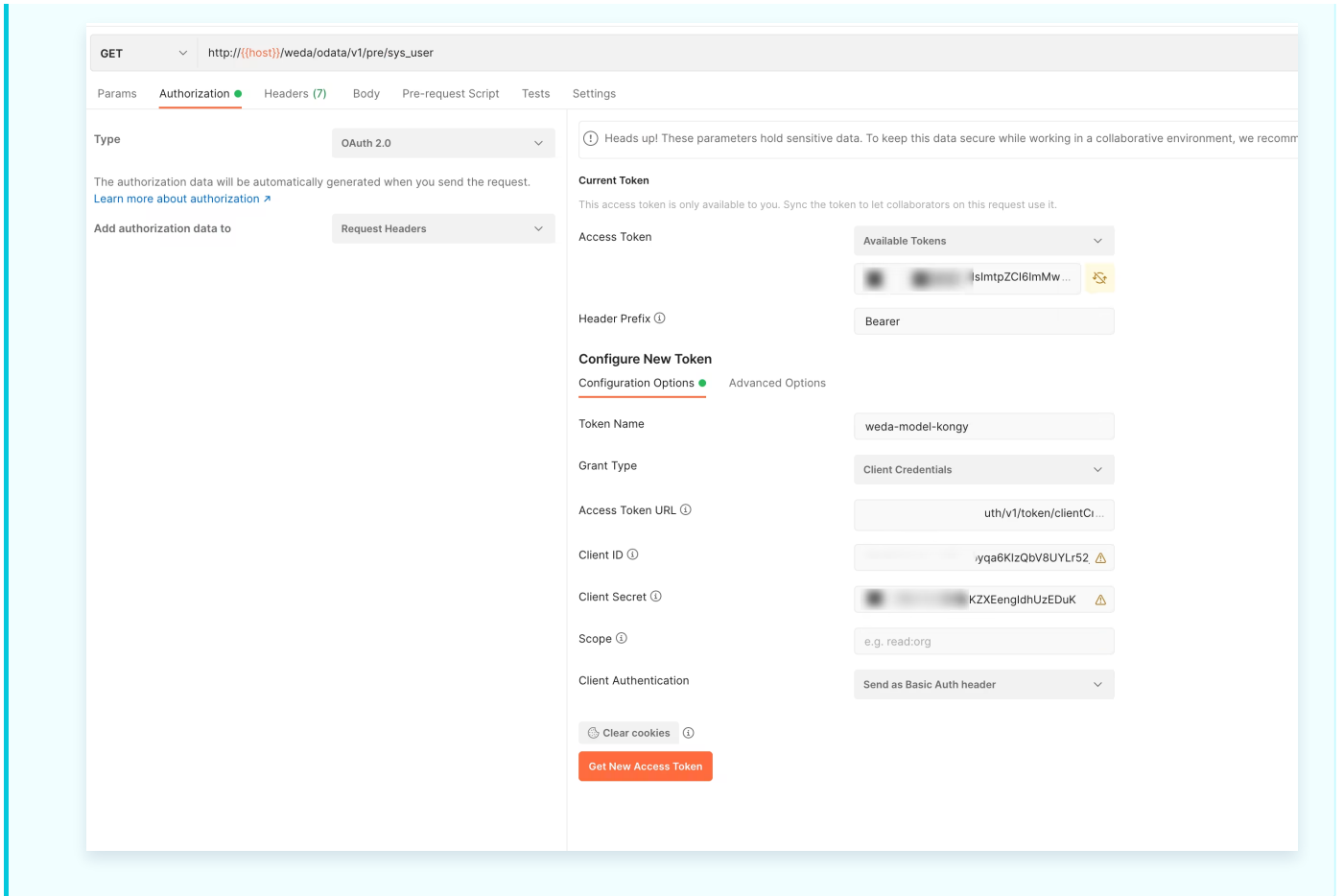
//查看记录
openApiClient.GET(token, envId, envType, datasourceName);
*/
}
}
```

快速体验：使用 Postman 调用开放接口

1. 打开 Postman 工具，添加一个 GET 请求。进入 **Authorization** 页面完成相应配置。

⚠ 注意：

Access Token URL 参数设置，需要在域名后添加 `/auth/v1/token/clientCredential`。示例：`https://lowcode-8g171waac4be77f6.ap-shanghai.tcb-api.tencentcloudapi.com/auth/v1/token/clientCredential`。



2. 进入 headers 页面，设置相应参数即可。

