云手机

串流加速服务



腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【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】

腾讯云

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

串流加速服务
概述
SDK 接入说明

版权所有：腾讯云计算（北京）有限责任公司

串流加速服务
概述

最近更新时间：2025-11-24 16:40:52

概念说明

依托腾讯云音视频能力和优质的专线网络，提供完整的云端编码、采集、串流与触摸闭环支持。该功能可以显著提高远距离访问云手机的流畅度，提升串流连接稳定性。

适用场景

- 跨境电商：通过接入串流加速 SDK，实时操作位于目标市场的云手机，处理订单和提供售前/售后服务。
- 社媒运营：运营人员可通过串流加速后低延迟的云手机，进行本地化的社交媒体内容发布、网红合作测评、用户互动。
- 多云统一管理：无论云手机资源部署在腾讯云、客户自有机房还是其他云厂商，均可通过本方案实现音视频串流能力的统一纳管，为企业提供一致、稳定、高清的远程访问体验，简化运维复杂度。



方案优势

- 基于腾讯云优质专线网络，保障连接低延迟、高稳定。
- 覆盖全球主流节点，在国内外主要地区均设有接入点，支持用户就近接入，有效降低访问延迟，提升全球用户体验。
- 由腾讯云手机音视频服务提供完整的云端编码、采集、串流与触摸闭环支持，降低客户端开发负担。
- 支持自定义数据通道与上行音视频数据回调，并通过本地通信实现高效数据交互。
- 提供功能完备的终端 SDK，覆盖建连、解码等关键环节，实现端到端闭环体验。

接入方式

方式一：使用腾讯云手机资源

集成客户端 SDK 和对接后台调度管理，参考 SDK 接入说明 。

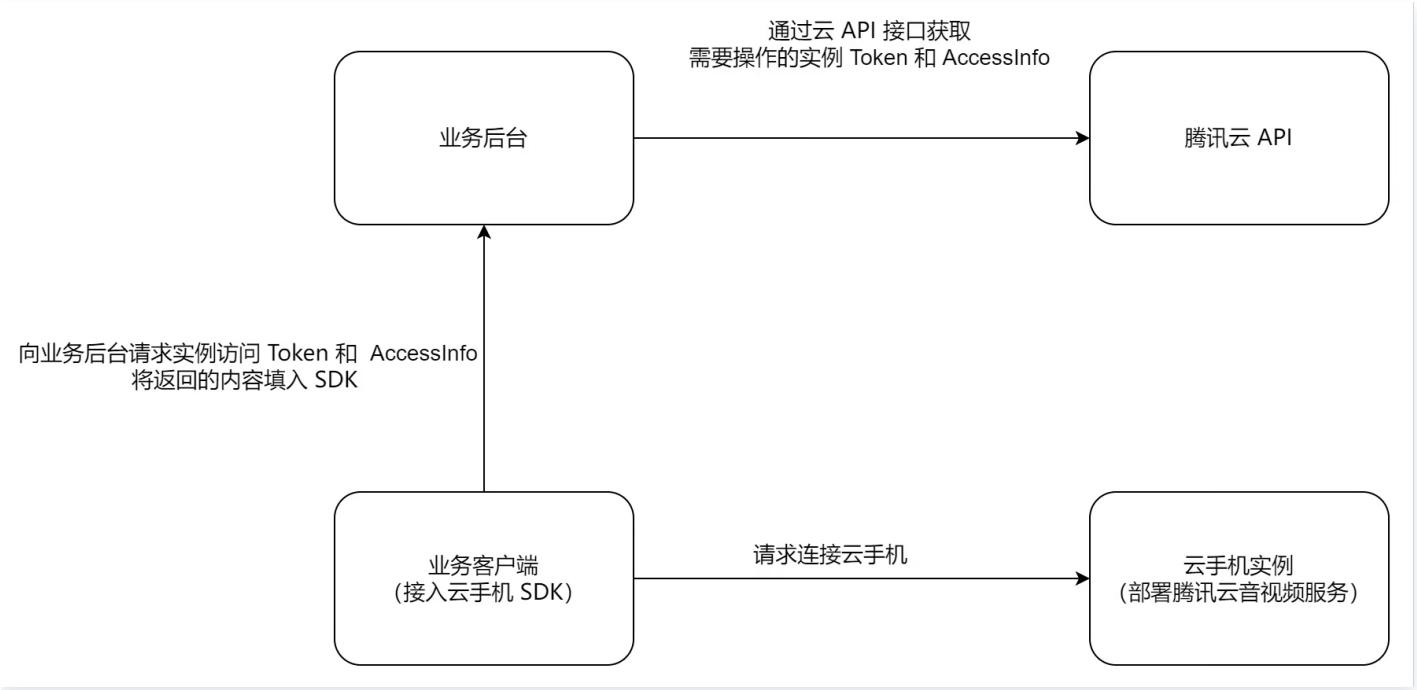
方式二：使用第三方资源

若接入的是第三方云手机实例，需集成客户端 SDK 和对接后台调度管理，还需要在第三方云手机实例上部署腾讯音视频服务。若您有相关接入需求，请联系您的腾讯云商务或 填写表单 。

SDK 接入说明

最近更新时间：2025-11-24 16:40:52

接入流程



如上图所示，云手机 SDK 的使用流程如下：

1. 通过业务后台调用云 API 获取需要在客户端中操作实例的访问 Token 及 AccessInfo。
   - 加速腾讯云手机实例使用 CreateAndroidInstancesAccessToken 获取
   - 加速第三方云手机实例使用 CreateAndroidInstanceAcceleratorToken 获取

> **说明：**
> 若接入的是第三方云手机实例，需要在第三方云手机实例上部署腾讯音视频服务。若您有相关接入需求，请联系您的腾讯云商务或 填写表单 。

2. 将获取的访问 Token 及 AccessInfo 下发给业务客户端，并调用 SDK 接口 setAccessToken 添加到云手机 SDK。
3. 业务客户端调用云手机 SDK 的接口实现连接云手机实例，具体接口及功能请参考 云手机 SDK 文档 。

Web 快速接入

获取 SDK

SDK	下载地址	SDK 说明文档	版本发布日志

JavaScript SDK	GitHub下载	接口说明	Release Notes

引入 SDK

> ⓘ **说明：**
> TCGSDK export UMD module 已默认将 TCGSDK 挂载到 Window 下，可直接使用 TCGSDK 对象，也可以自行初始化实例。SDK 说明可参考 SDK 声明文件 。

方法一： script 标签引入。

```
<script src="path/to/sdk" charset="utf-8"></script>
```

方法二： 模块形式引入，默认 export module 为 UMD。

```
import TCGSDK, {CloudGamingWebSDK} from 'path-to-TCGSDK';

// 直接使用 TCGSDK 或 new CloudGamingWebSDK()
```

启动 SDK

```
// 以下均为伪代码

// 调用云 API 创建安卓实例访问Token
// 加速腾讯云手机实例参考接口文档：
https://cloud.tencent.com/document/api/1162/119708
// 加速第三方云手机实例参考接口文档：
https://cloud.tencent.com/document/product/1162/125484

// 加速腾讯云手机实例可以参考 https://github.com/tencentyun/cloudgame-js-
sdk/blob/master/samples/android_instance/src/android_instance/network.ts
const {AccessInfo, Token} = await
CreateAndroidInstancesAccessToken({AndroidInstanceIds});

TCGSDK.init({
  appid: 1234567,
  mount: 'mount-point', // 填入需要挂载入云手机视图的节点
  userId: 'userA',      // 用户id，建议业务自定义，选填
  accessToken: {
    accessInfo: AccessInfo,
```

```
    token: Token
  },
  debugSetting: {
    showLog: true,
  },
  // 连接成功回调
  onConnectSuccess: (res) => {
    console.log('onConnectSuccess', res);
  },
  // 网络中断/被踢触发此回调
  onDisconnect: (res) => {
    console.log('onDisconnect', res);
  },
  // 初始化成功回调
  onInitSuccess: async (res) => {
    console.log('%c onInitSuccess', 'color: red', res);

    // 开始渲染
    TCGSDK.access({
      instanceId: AndroidInstanceIds[0]
    });
  }
});
```

完整 Demo

完整代码可参考 Android Instance React Demo 。

Android 快速接入

获取 SDK

SDK	下载地址	SDK 说明文档	版本发布日志
Android SDK	通过 Maven 集成	接口说明	Release notes

SDK 集成

在应用模块的 'build.gradle' 中引用：

```
implementation 'com.tencent.tcr:tcrsdk-full:3.26.1'
```

SDK 使用

```
// 请求云手机实例的访问信息AceessInfo和鉴权Token
private void requestAccessToken() {

AsyncCallback<ExpServerResponse<CreateAndroidInstancesAccessTokenRespons
e>> createAndroidInstancesAccessTokenCallback = new
AsyncCallback<ExpServerResponse<CreateAndroidInstancesAccessTokenRespons
e>>() {
        @Override
        public void
onSuccess(ExpServerResponse<CreateAndroidInstancesAccessTokenResponse>
expServerResponse) {
            if (expServerResponse.Error != null) {
                Log.e(TAG, "CreateAndroidInstancesAccessToken
expServerResponse.Error: " + expServerResponse.Error);
                return;
            }
            if (expServerResponse.Response == null) {
                Log.e(TAG, "CreateAndroidInstancesAccessToken Response:
null");
                return;
            }

            // 初始化 TcrSdk
            initTcrSdk(expServerResponse);
        }

        @Override
        public void onFailure(int code, String errorMsg) {
            Log.e(TAG, "CreateAndroidInstancesAccessToken failed: " +
code + ", " + errorMsg);
            Toast.makeText(FunctionActivity.this, "获取实例访问信息失败: " +
code + ", " + errorMsg, Toast.LENGTH_LONG).show();
            finish();
        }
    };
    ......
}
```

```
// 初始化TcrSdk
private void
initTcrSdk(ExpServerResponse<CreateAndroidInstancesAccessTokenResponse>
expServerResponse) {
    TcrSdk.TcrConfig config = new TcrSdk.TcrConfig();
    config.type = SdkType.CloudPhone;
    config.ctx = this;
    config.logger = new TcrLogger() {
        // 记录日志
    };
    config.callback = new AsyncCallback<Void>() {
        @Override
        public void onSuccess(Void result) {
            Log.i(TAG, "init TcrSdk success");
        }

        @Override
        public void onFailure(int code, String msg) {
            String errorMsg = "init TcrSdk failed:" + code + " msg:" +
msg;
            Log.e(TAG, "init TcrSdk fail. errorMsg: " + errorMsg);
        }
    };
    config.accessInfo = expServerResponse.Response.AccessInfo;
    config.token = expServerResponse.Response.Token;
    TcrSdk.getInstance().init(config);
}

// 创建Tcr会话对象
private void initTcrSession() {
    TcrSessionConfig tcrSessionConfig =
TcrSessionConfig.builder().observer(mSessionObserver).build();
    mTcrSession =
TcrSdk.getInstance().createTcrSession(tcrSessionConfig);
    if (mTcrSession == null) {
        Log.e(TAG, "mTcrSession = null");
        showToast("创建会话失败，请检查TcrSdk是否初始化成功",
Toast.LENGTH_SHORT);
    }
```

```
}

// 创建渲染视图
private void initTcrRenderView() {
    if (mTcrSession == null) {
        showToast("创建渲染视图必须有关联的会话", Toast.LENGTH_SHORT);
        return;
    }
    // 创建渲染视图
    mRenderView = TcrSdk.getInstance().createTcrRenderView(this,
mTcrSession, TcrRenderViewType.SURFACE);
    if (mRenderView == null) {
        Log.e(TAG, "mRenderView = null");
        showToast("创建渲染失败，请检查TcrSdk是否初始化成功",
Toast.LENGTH_SHORT);
        return;
    }
    // 将渲染视图添加到界面上
    ((FrameLayout)
findViewById(R.id.render_view_parent)).addView(mRenderView);
}

// Tcr会话的观察者，处理各类事件通知的消息和数据
private final Observer mSessionObserver = new Observer() {
    @Override
    public void onEvent(TcrSession.Event event, Object eventData) {
        switch (event) {
            case STATE_INITED:// 本地会话对象初始化完成
                // 可以开始连接指定的云手机。
                boolean ret = mTcrSession.access(mGroupInstanceIds,
mIsGroup);
                if (!ret) {
                    showToast("连接云手机失败，请重试", Toast.LENGTH_SHORT);
                    finish();
                }
                break;
            case STATE_CONNECTED:// 成功和指定的云手机建立连接
                // 设置云手机的主控、同步列表、请求主控视频流
                if (mIsGroup) {
                    String masterId = mGroupInstanceIds.get(0);
```

```
TcrSdk.getInstance().getAndroidInstance().setMaster(masterId);

TcrSdk.getInstance().getAndroidInstance().setSyncList(mGroupInstanceIds)
;

TcrSdk.getInstance().getAndroidInstance().requestStream(masterId,
"open", "low");
                }
                break;
            case STATE_RECONNECTING:
                showToast("重连中...", Toast.LENGTH_LONG);
                break;
            case STATE_CLOSED:
                showToast("会话关闭", Toast.LENGTH_SHORT);
                finish();
                break;
            default:
                break;
        }
    }
};
```

完整 Demo

完整接入代码可参考 TcrCloudPhoneDemo 。