WEB UPhone - SDK 接入文档

变更历史

版本	发版日期	更新内容
1.0.0	2022/6/23	1. 初始版本
1.0.1	2022/7/30	1. 修改 sdk 目录结构

目录

1 概述

欢迎使用 WEB UPhone SDK，产品能够为开发者提供更便捷接入、高可靠的云手机服务，让开发者快速搭建实时项目。开发者可在 WEB UPhone SDK 提供的功能基础上开发新功能，同时还可结合官方提供的配套 demo 进一步了解内置功能，应用场景广阔，开发简单易懂。

2 快速入门&集成 SDK

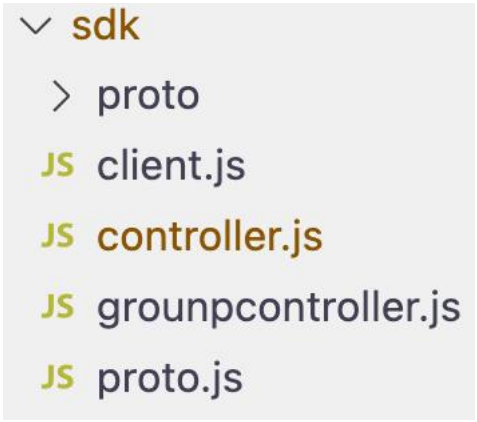
2.1 准备环境

在开始集成 WEB UPhone SDK 前，请确保开发环境满足一下要求：

- 准备一台可以连接到互联网的 Windows 或 macOS 计算机。
- 使用最新版本的 Chrome 浏览器。

2.2 导入 SDK

1. 解压 SDK 压缩包，将 sdk 文件夹放入项目中。*开发者可借鉴官方提供 demo。



2. 用户创建媒体界面，并在媒体界面导入"proto.js"、"controller.js"。

```
<script src="./sdk/proto.js"></script>
```

```
  import control from "./sdk/controller.js"
```

说明：在创建引擎之前，开发者需定义一个 id 为 remote-video 的 video 标签，SDK 会在这个元素内加载音视频。video 标签设置 autoplay、playsinline、webkit-playsinline 等属性。

示例代码

```
  <div>

      <video id="remote-video" autoplay="autoplay" playsinline webkit-playsinline><
/video>

    </div>

<script src="./sdk/proto.js"></script>

<script type="module">

import controller from "../sdk/controller.js ";

</script>
```

3. 初始化创建引擎实例，创建 controller 引擎实例，将参数传入。

var Controller = new controller(loadingParams)

示例代码

```
  <div>

      <video id="remote-video" autoplay="autoplay" playsinline webkit-playsinlin
e></video>

    </div>f

<script src="./sdk/proto.js"></script>

<script type="module">

import controller from "../sdk/controller.js "; //文件路径不固定

//----第一步初始化 SDK----

var loadingParams = {

        Id: "替换为可用 ID",

        mediaConstraints: {

            audio: true,

            video: true,

        },

        gamename: "游戏包名",

        jobid: "唯一即可",

        token:"如果调用 api 接口 SetUPhoneToken 进行了设置，此处为必填，否则为可选"

    }

 var Controller = new controller(loadingParams)

</script>
```

参数说明

Id:可连接的云手机 ID。必填

mediaConstraints:连接约束条件。必填

audio:true/false,为 true 则获取音频流。

video:true/false,为 true 则获取视频流

gamename:游戏包名（可选参数）

jobid:唯一即可（可选参数）

*token:如需调用 api 接口 SetUPhoneToken 进行了设置,此处为必填。

3 API 接口

3.1 状态回调函数

具体例子请查看官方 demo

示例代码

```
Controller.onstatus(statustype,callback)
```

参数	类型	必填	意义
callback	回调函数	(states)=>{}	states 状态回调函数返回值
statustype	string	是	可选值 devicestatus: 获取设备状态 networkstatus: 获取网络连接状态 gamestatus: 获取一键启动游戏状态

```
    //设备连接状态回调
    Controller.onstatus("devicestatus", (states) => {
      if (states == 1001) {
        console.log("连接设备失败");
      }
      if (states == 1003) {
        console.log("创建设备控制失败");
      }
      if (states == 1008) {
        console.log("服务器应答失败");
      }
      if (states == 1026) {
        console.log("设备已被占用");
      }
      if (states == 73002) {
       console.log("设备不存在");
      }
    });
    //网络连接状态回调
    Controller.onstatus("networkstatus", (states) => {
```

```
    if (states == "connected") {
      console.log("网络连接成功");
    }
    if (states == "disconnected") {
      console.log("网络连接失败");
    }
  });
  //启动游戏状态回调
  Controller.onstatus("gamestatus", (states) => {
    if (states == "success") {
      console.log("启动成功");
    }
    if (states == "fail") {
      console.log("启动失败");
    }
  });
  //切换分辨率状态回调
  Controller.onstatus("resolution", (states) => {
    if (states == "success") {
      console.log("切换成功");
    } else {
      console.log("切换失败");
    }
  });
```

3.2 连接云手机

开发者在媒体界面创建实例之后，即可启动云手机建立连接。

示例代码

```
Controller.startConnection();
```

3.3 获取云手机媒体流

建立信令服务器连接并且 p2p 连接成功后，即可获取远程媒体流。

示例代码

```
Controller.getRemoteStream()
```

```
var stream = Controller.getRemoteStream();

var phoneVideo = document.getElementById("remote-video");

phoneVideo.srcObject = stream;      // 添加视频流到 video 标签
```

说明：目前云手机视频流比例分为 1:2 或 9:16 两种。开发者可在 video 标签方法 onloadedmetadata 中 获取到视频流实际宽高，根据视频比例设置需要展示的 video 大小即可。

如：获取到视频流宽为 720，高为 1440。video 标签可设置宽高保证 1:2 比例即可。

3.4 断开云手机连接

断开连接调用以下接口

示例代码

```
Controller.closeConnection();
```

3.5 重新连接

重新建立云手机连接

示例代码

```
Controller.reStart();
```

3.6 切换分辨率

切换分辨率时传入规定好的分辨率 id，即可切换对应的分辨率

说明：视频流比例不同，切换分辨率传递 id 不同，可参照官方 demo 说明。

示例代码

```
Controller.changeResolution(id);
```

参数	类型	必填	意义
id	number	是	1:2 视频流比例： 1 标清 2 高清 3 超清 9:16 视频流比例： 0 标清 3 高清 6 超清

3.7 获取最后一次操作的时间戳

示例代码

```
Controller.getLastTimeStamp(); // 单位毫秒
```

3.8 获取延迟显示

获取网络延迟时间

示例代码

```
Controller.getNetDelay(callback)
```

```
Controller.getNetDelay((roundtime) => {
          console.log(roundtime) //单位毫秒
});
```

参数	类型	必填	意义
callback	回调函数	(roundtime)=>{}	延迟回调返回延迟信息 单位毫秒

3.9 返回云手机桌面

返回到云手机桌面

示例代码

```
Controller.backHome();
```

3.10 一键拉起游戏

调用该方法一键启动云手机内游戏（须在设备连接成功以后调用方可生效）

示例代码

```
//一键启动游戏
  let message = {
      gamename:"xxx.xxx.xxx", //游戏包名
     jobid: "", //后台区分任务标识，唯一即可
  };
  Controller.startGame(message);
```

3.11 清理云手机后台进程

清理云手机后台应用（前端显示的应用不受影响）

示例代码

```
Controller.clearUp()
```

4 导入官方 Demo

1. 下载官方 demo 压缩包，解压文件将文件夹导入到开发工具。
2. 可在本地服务环境下运行，浏览器调试。

5 常见错误码

0：成功

1000:"初始化连接失败";

1001:"连接设备失败";

1002:"初始化编码实例失败";

1003:"创建设备控制失败";

1008:"创建远端媒体响应失败";

1009:"容器内部错误";

1025:"参数无效";

1026: "设备被占用";

1027: "生成本地媒体信息超时";

1028: "收集远端网络信息超时";

1029: "生成远端媒体信息失败";

1101：未知错误

1102:网络连接超时

1103:网络连接断开

1104:网络连接关闭

1105:网络连接错误

42100：请求参数错误

50000：服务器内部错误

73002：设备不存在

90010：发送本地媒体信息错误

90011：发送本地网络信息错误

90012：未找到远端媒体信息

90013：未找到远端网络信息

211705:"应用未找到"