游戏盾 SDK
用户使用手册

目 录

1. 手册目的.....3
2. 读者范围.....3
3. 名词定义.....3
4. 产品介绍.....4
4.1. 产品简介.....4
4.2. 运行环境.....4
5. 游戏盾 SDK 控制台使用说明.....4
5.1. 新购版本.....4
5.2. 用户登录.....4
5.2.1. 登录.....4
5.2.2. 浏览器版本升级提醒.....4
5.2.3. 安全退出页面.....4
5.3. SDK 管理.....5
5.3.1. 产品名称备注.....5
5.3.2. 使用手册下载.....6
5.3.3. 管理.....6
5.3.4. 监控报表.....10
5.3.5. 新购.....11
5.3.6. 下载 SDK.....11
5.4. SDK 接入具体操作.....12
5.4.1. Windows 接入.....12
5.4.2. Android 接入.....13
5.4.3. iOS 接入.....16

1. 手册目的

使游戏盾 SDK 的用户全面了解各项功能的使用方法。

2. 读者范围

适用于使用游戏盾 SDK 产品的用户。

3. 名词定义

名词	定　义

4. 产品介绍

4.1. 产品简介

游戏盾 SDK 提供内含 windows、安卓、IOS 版本的 SDK 下载，通过接入 SDK，由 SDK 接管所有的通信流量，进行调度和加密传输，满足抗 D、防 C、流量加密等业务需求，为用户提供优质的网络环境，游戏极速畅通无阻。

4.2. 运行环境

操作系统要求：支持 Windows、安卓、IOS 系统。

5. 游戏盾 SDK 控制台使用说明

5.1. 新购版本

使用游戏盾 SDK 包月/年资源前，您需要按月/年开通游戏盾 SDK 产品。游戏盾 SDK 产品的包月/年单价由您选择的产品套餐版本和时效决定。

5.2. 用户登录

5.2.1. 登录

登录可通过购买后华为云市场提供的链接进行跳转登录。

5.2.2. 浏览器版本升级提醒

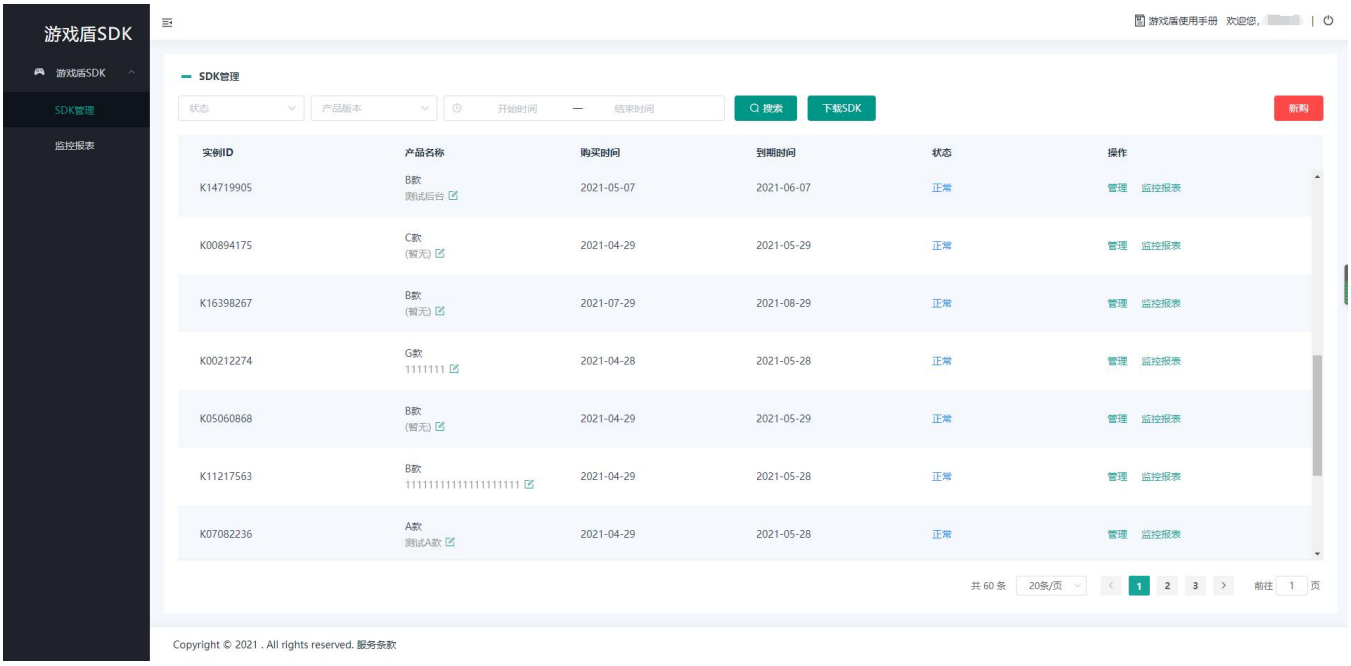
游戏盾 SDK 满足大部分浏览器需求。但目前 IE6.0 和 IE7.0 版本浏览器对于诸多网站访问的安全要求及速度要求已经无法满足，为了让你获得更加安全快速的体验，建议您使用 IE9.0 及以上的浏览器。

5.2.3. 安全退出页面

可通过右上角“安全退出”按钮安全退出游戏盾 SDK 的登录，。

5.3. SDK 管理

游戏盾 SDK 界面默认采用左右结构，左侧为导航菜单；右侧为产品管理模块。当前页面显示如下图所示：

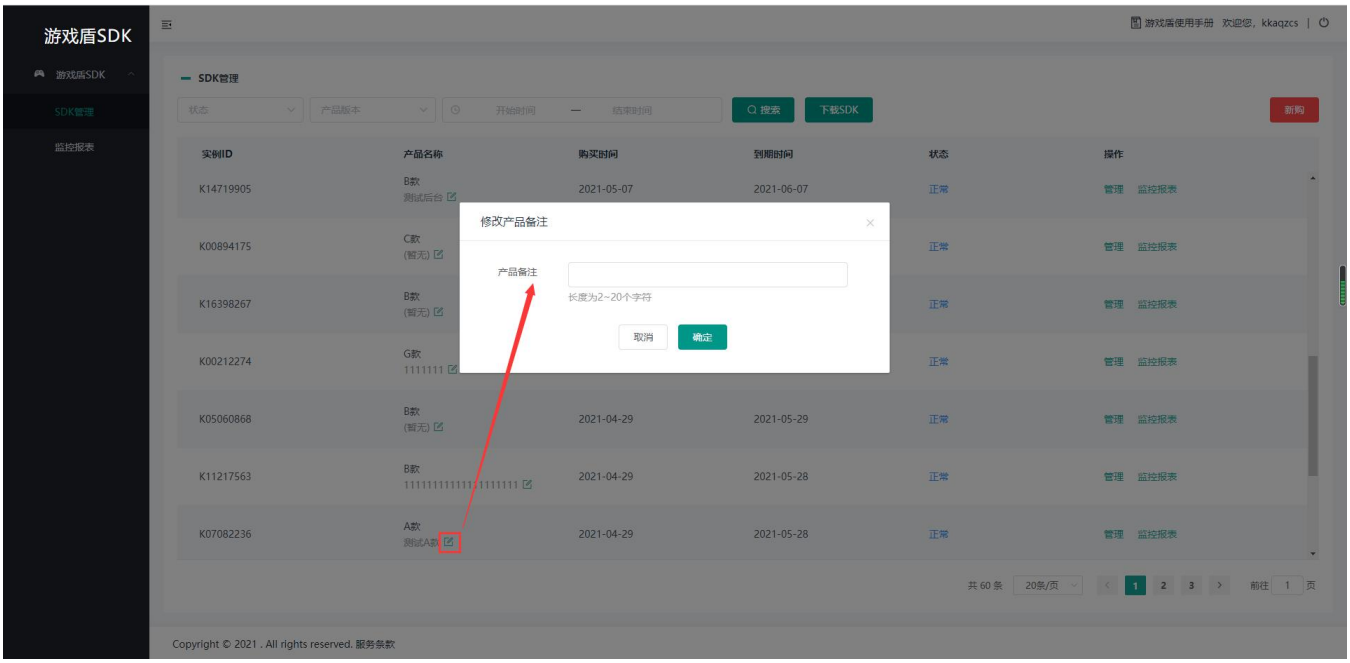


此页面显示用户购买游戏盾 SDK 的所有产品。该列表中包括了实例 ID、产品名称、购买时间、到期时间、状态。用户可通过列表上方的状态、产品版本、开始时间、结束时间搜索出对应的产品信息。

在此列表中，用户可针对其购买的产品进行查管理、查看监控报表（带宽使用趋势及连接数趋势）。

5.3.1. 产品名称备注

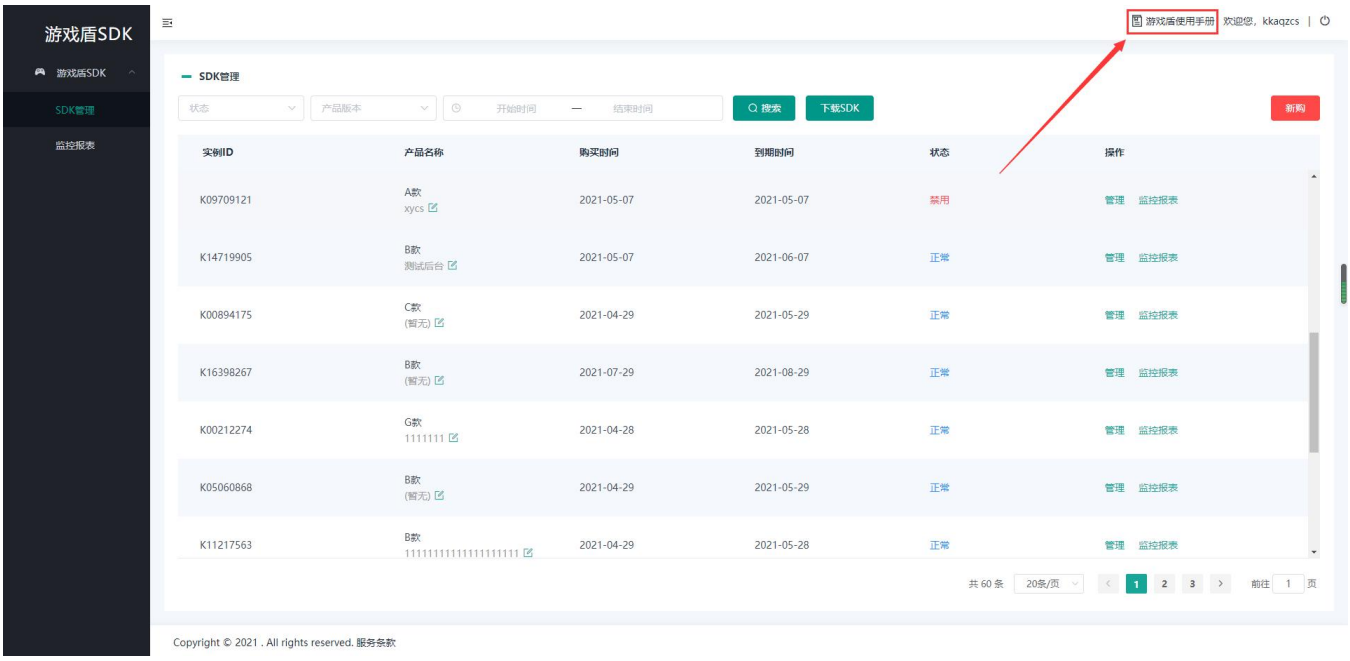
当用户购买多个游戏盾 SDK 产品时，为了方便区分可通过点击列表中【产品名称】旁边的 按钮进行产品备注，如下图所示：



注：产品名称备注不得超过 20 个字符。

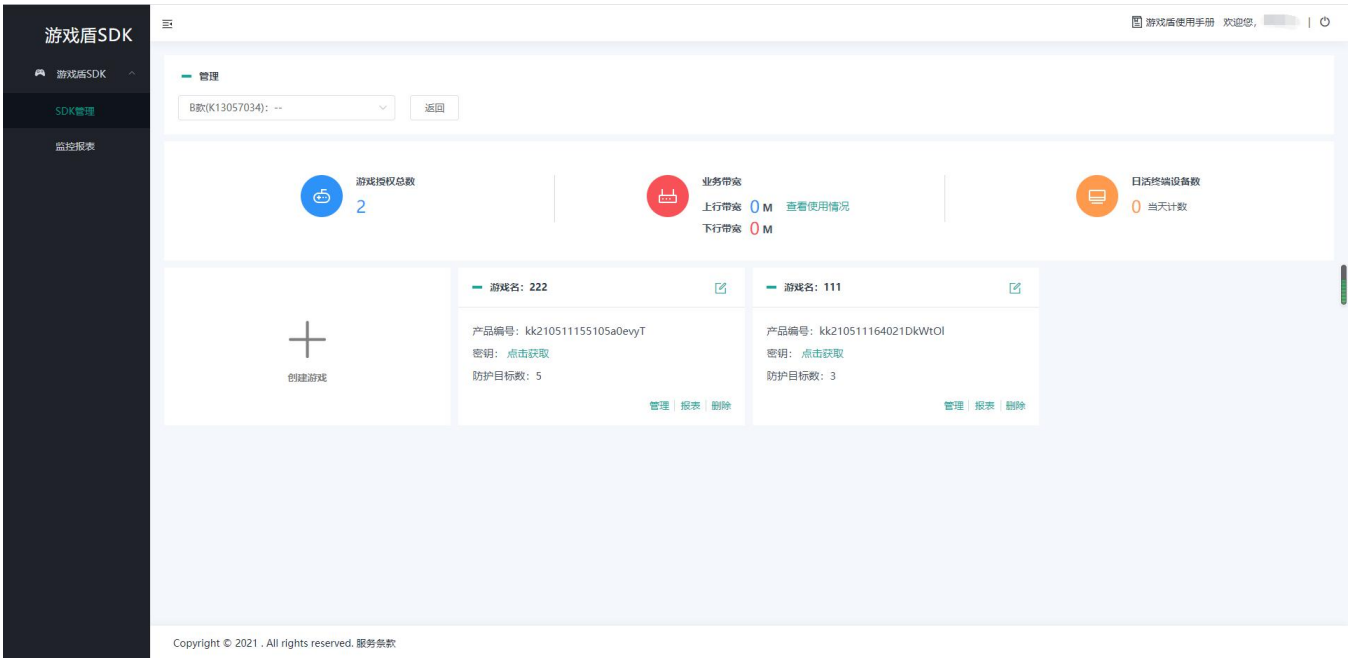
5.3.2. 使用手册下载

可点击右上角游戏盾使用手册进行查看、下载使用手册。



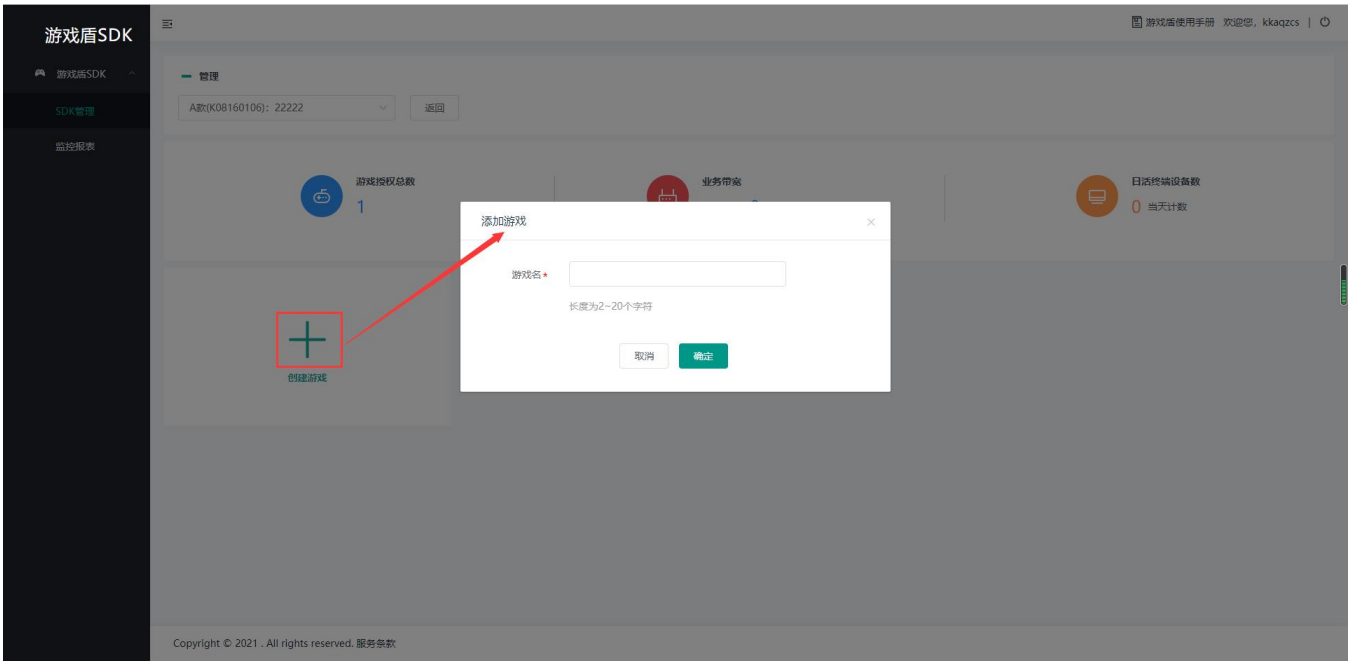
5.3.3. 管理

购买不同产品名称的产品，其产品参数有所区别。用户点击列表中【管理】按钮，可查看此产品对应的详细参数：包括游戏授权总数、业务带宽、日活终端设备数、该产品名称对应的游戏等。并可对该产品下对应的游戏进行配置及报表查看。日活终端设备数即当天的玩家数量统计，相同玩家同一天多次登录的话只计算一次。



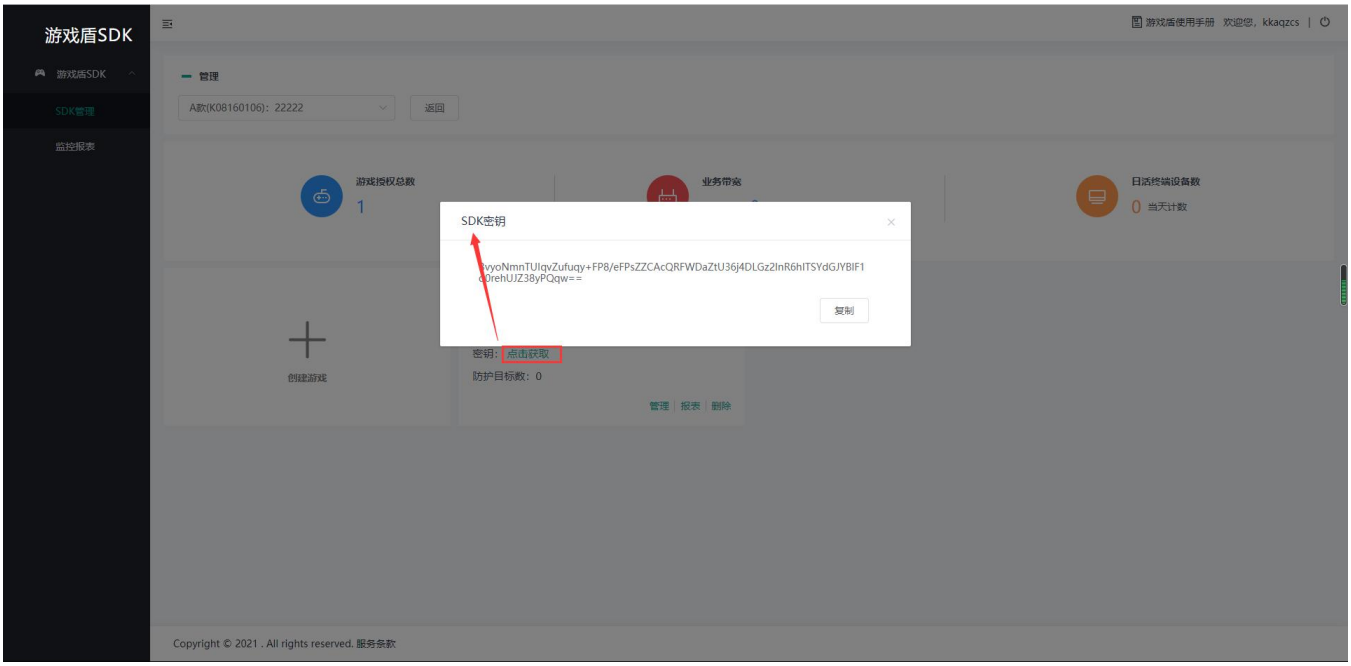
5.3.3.1. 创建游戏

根据购买的产品名称可看到对应的的游戏授权数，有几个游戏授权数就说明可以创建多少个游戏。每个游戏都有对应的产品编号及密钥，也可对单个游戏进行转发规则配置。



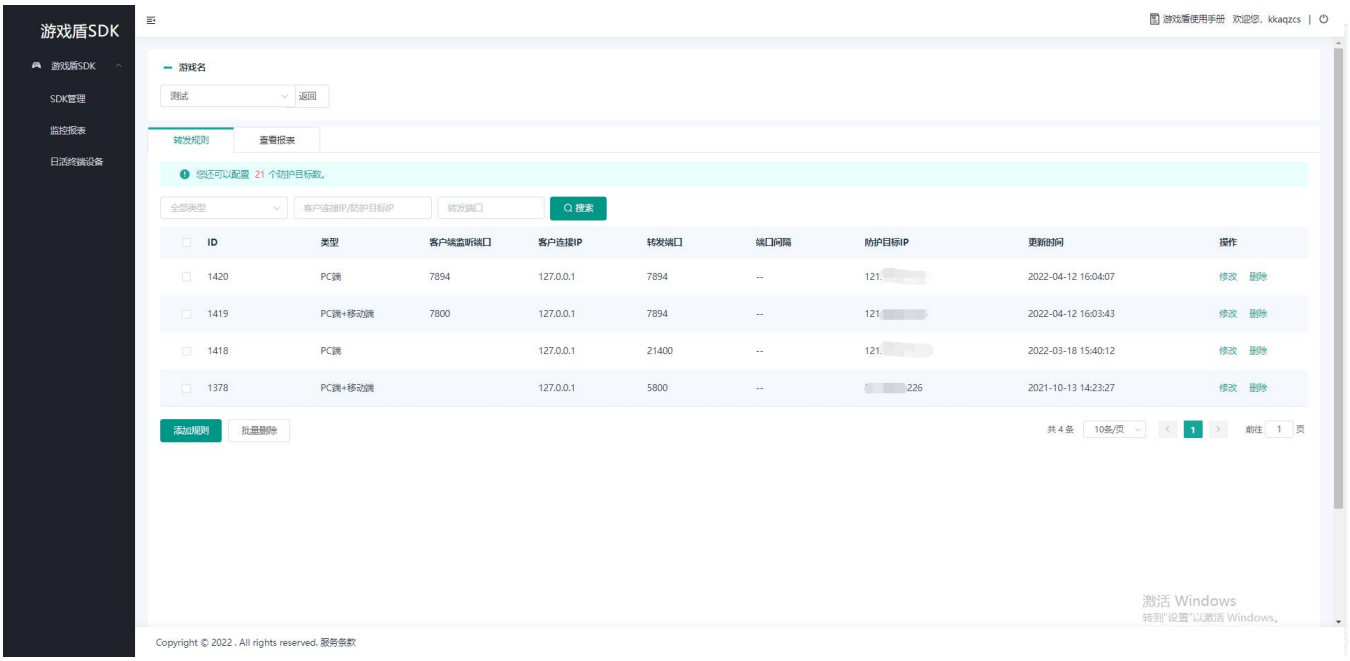
5.3.3.2. 获取密钥

用户购买游戏盾 SDK 后需要下载 SDK 软件安装，并对其游戏程序进行相应修改，通过密钥进行连接，每款游戏都有对应的密钥。SDK 接入具体操作详见【5.4】。



5.3.3.3. 游戏-管理-转发规则

可对每个游戏进行转发规则的配置，在转发规则设置页面，可对该游戏进行规则的添加删除修改，若用户的规则设置较多时，可通过列表上方的类型、客户连接 IP、防护目标 IP 进行搜索，同时可进行游戏名的切换，查看不同游戏下的规则设置。支持自定义转发端口可转发至另一个端口。



若用户需要对当前规则进行修改时，可找到对应规则，点击列表右边的【修改】按钮进行修改。修改保存后，程序将默认通过修改后的规则执行。同时用户可对不需要的规则进行删除，支持批量删除。

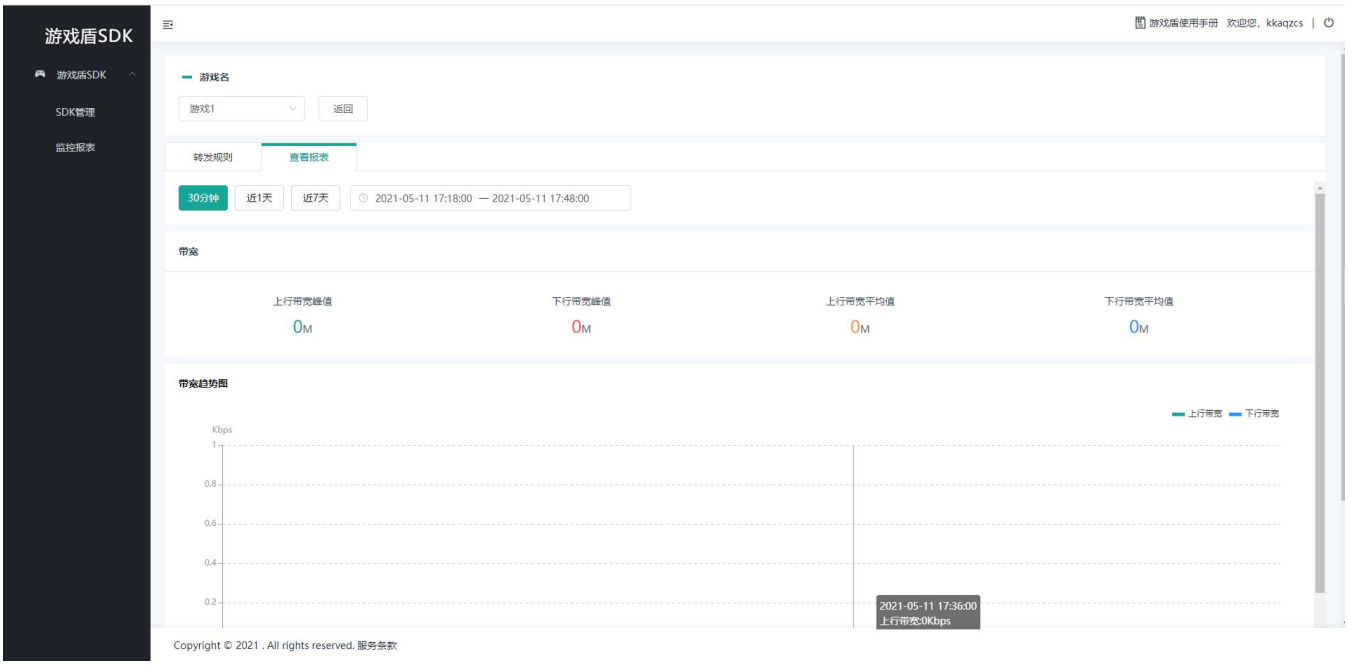
添加规则时，用户需要选择使用场景是 PC 端、移动端或者 PC 端+移动端。若类型选择 PC 端，则用户连接 IP 提供了 6 种供其选择，若类型选择移动端，则客户连接 IP 默认为 127.0.0.1。转发端口用户可自行填写，但建议使用冷门端口，避免端口冲突影响到产品使用。当转发端口需要批量添加时，可点击【批量】按钮，即可填写开始端口至结束端口，端口间隔默认为 1，用户可对间隔进行修改，代表每个端口之间的差值。防护目标 IP 客户需要自行添加，默认一条规则对应一个防护目标 IP。

5.3.3.4. 游戏-管理-源站业务 IP 修改

规则添加完成，用户需要将其源站 IP 中的业务 IP 更改成规则中的客户连接 IP。

5.3.3.5. 游戏-管理-查看报表

用户使用游戏盾 SDK 期间，控制台将对各产品对应的游戏使用的带宽及连接数的走势形成图表，方便客户分析查看。图表默认展示近 30 分钟的趋势，最多支持近 7 天的趋势查看。如该产品对应多个游戏，可通过切换左上方的游戏名进行查看。

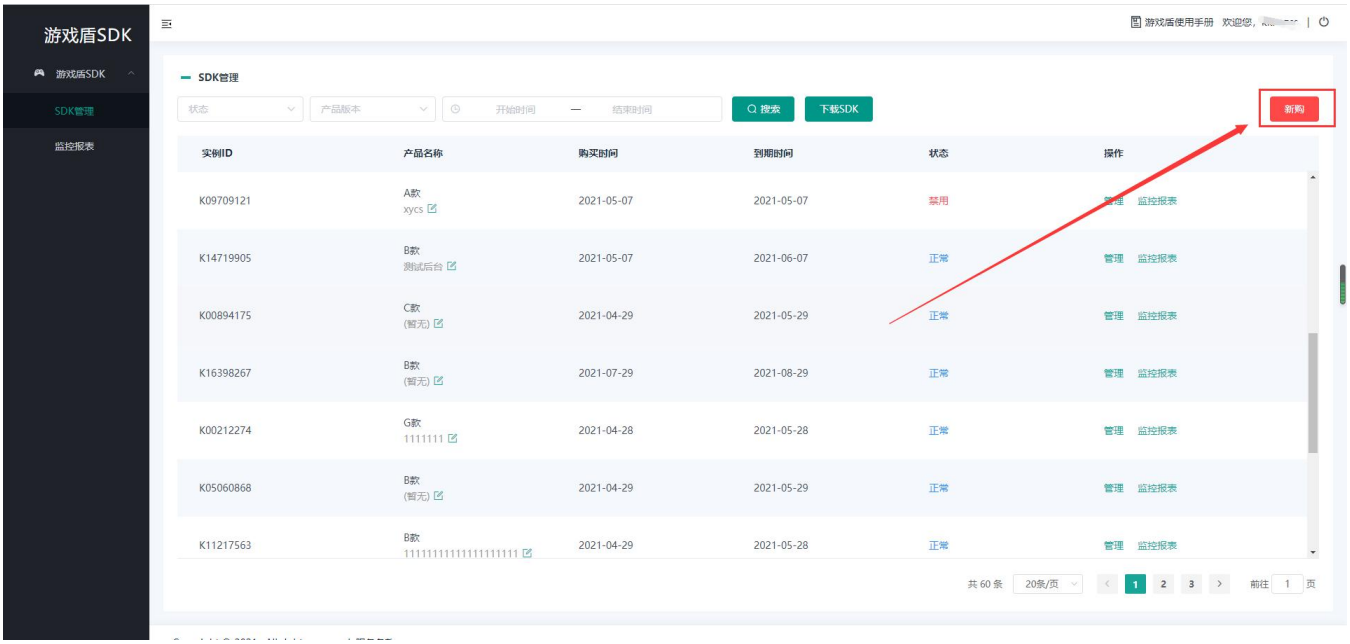


5.3.4. 监控报表

此处的监控报表是用于查看各产品使用的带宽及连接数的走势图表，图表默认展示近 30 分钟的趋势，最多支持近 7 天的趋势查看。如该客户购买多个产品，可通过切换左上方的产品名称进行查看。

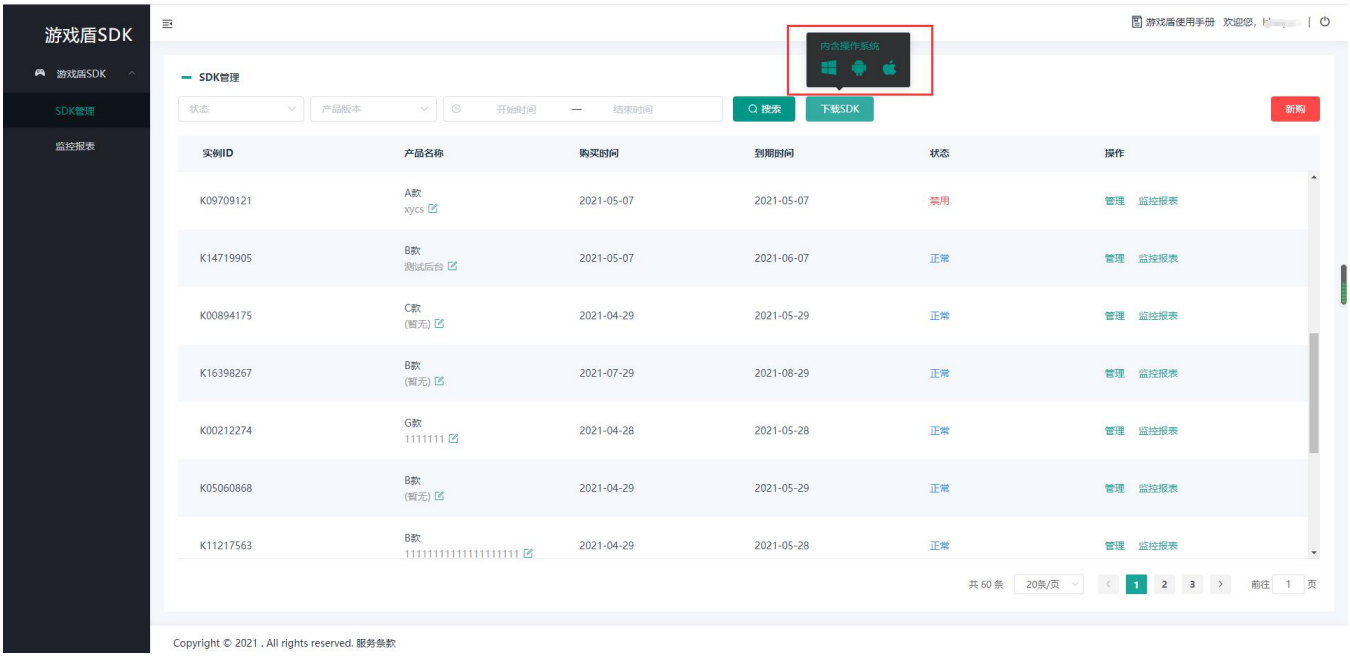
5.3.5. 新购

用户点击【SDK 管理】页面上【新购】按钮，则跳转至游戏盾 SDK 的官方网站，网站中可选择需要购买的产品版本。



5.3.6. 下载 SDK

用户购买了游戏盾 SDK，需要下载 SDK 软件进行接入。在【SDK 管理】页面点击【下载 SDK】即可。当前 SDK 已支持 Windows、安卓、IOS 系统。



5.4. SDK 接入具体操作

前提条件：

1、完成目标产品的转发规则配置。（可由技术支持协助配置）

2、从游戏盾 SDK 管理页面下载 SDK 安装包，并获取各个版本的 SDK 包。

3、从游戏盾 SDK 管理列表获取目标产品的 SDK 密钥。

客户端三端接入流程：

1、客户端中关于网络请求的地址改为本地地址而不是直接请求，具体地址从后台获取。

2、客户端代码初始化入口调用 start 接口。

5.4.1. Windows 接入

介绍通过 Windows 接入游戏盾 SDK 的操作方法

前提条件：

1、完成目标产品的转发规则配置。

2、从游戏盾 SDK 管理页面下载 SDK 安装包，并获取 Windows 版本的 SDK 包。

3、从游戏盾 SDK 管理列表获取目标产品的 SDK 密钥。

文件说明：

文件名	描述
yjs_sdk\lib\windows\CloudClinkAPITest	示例工程文件
yjs_sdk\lib\windows\x86\CloudClinkAPI.dll	32 位的 dll
yjs_sdk\lib\windows\x64\CloudClinkAPI.dll	64 位的 dll
yjs_sdk\lib\windows\CloudClinkAPI.h	接口头文件

接口说明：

```
/*****************************************************************/
/*
 * 函数名称: ClinkStart
 * 输入参数: sdk配置密钥
 * 输出参数: 无
 * 返 回 值: 0:成功、1: key错误、2.key过期、3.已存在相同调用（调用成功）、4.其他错误、5.无节点可用
 * 功能描述: 启动客户端安全接入组件(只能启动一次)
 */
/*****************************************************************/
extern "C" CLOUDCLINKAPI_API int ClinkStart(const char * key);

/*****************************************************************/
/*
 * 函数名称: ClinkStop
 * 输入参数: 无
 * 输出参数: 无
 * 返 回 值: 无
 * 功能描述: 停止，停止后只有进程重启后才可以再重新调用clinkStart函数，如果直接调用会出错
 */
/*****************************************************************/
extern "C" CLOUDCLINKAPI_API void ClinkStop();


/*****************************************************************/
/*
 * 函数名称: GetLocalIp
 * 输入参数: 无
 * 输出参数: 无
 * 返 回 值: 返回空表示获取错误
 * 功能描述: 获取本机ip，调用ClinkStart之后才能获取成功
 */
/*****************************************************************/
extern "C" CLOUDCLINKAPI_API const char* GetLocalIp();
```

注意：

1.要先调用 ClinkStart 再开启程序线程，在应用退出前一定要先调用 ClinkStop，这样可以关闭所有已建立的连接并释放资源。

2.GetLocalIp 提供了客户端本地 ip，方便用户自己传输请求的真实 ip。

示例代码：

```
#include <iostream>
#include <stdio.h>
#include "../../../../lib/windows/CloudClinkAPI.h"                    //包含api头文件

//dll要从对应的目录拷贝到exe相应目录下
#define x86   //编译64位需要注释,64为编译需要使用release版本
#ifdef x86
    #pragma comment(lib, "../../../../lib/windows/x86/CloudClinkAPI.lib")//链接32位api的lib文件
#else
    #pragma comment(lib, "../../../../lib/windows/x64/CloudClinkAPI.lib")//链接64位api的lib文件
#endif

using namespace std;

int main()
{
    /*-------------------------------------------------------------------------------
    注意：1.根据你的编译方式使用相应版本的CloudClinkAPI.dll文件
          2.编译后拷贝相应版本的“clinkAPI.dll”文件拷贝到exe的目录下
    -------------------------------------------------------------------------------*/
    std::cout << "连接开始" << endl;
    //定义sdk密钥
    char* cfg = "...";                                  //从管理界面获取的key
    //启动客户端安全接入组件(只能启动一次，哪怕是已调用了ClinkStop()也无法再重启,在应用启动后只能调用一次)，返回0表示成功，其它的为失败。
    //0：成功、1：代表验证串解析失败、2：序列号错误或过期（序列号验证）、3：已存在相同的终端（互斥）、4：其他错误
    int ret = ClinkStart(cfg);
    if (ret == 0)
    {
        cout << "连接成功。ret=" <<ret<< endl;
    }
    else
    {
        cout << "连接失败。ret=" << ret << endl;
    }
    system("pause");
    return 0;
}
```

5.4.2. Android 接入

介绍通过 Android 接入游戏盾 SDK 的操作方法

前提条件：

1、完成目标产品的转发规则配置。

2、从游戏盾 SDK 管理页面下载 SDK 安装包，并获取 Android 版本的 SDK 包。

3、从游戏盾 SDK 管理列表获取目标产品的 SDK 密钥。

文件说明：

文件名	描述
yjs_sdk\lib*文件夹	各个架构需要的库文件
yjs_sdk\lib\android\cloudclink.jar	接口头文件

接口说明：

```
/*****************************************************************/
/*
 * 函数名称：Start
 * 输入参数：sdk配置密钥
 * 输出参数：无
 * 返 回 值：0:成功、1：key错误、2.key过期、3.已存在相同调用（调用成功）、4.其他错误、5.无节点可用
 * 功能描述：开启程序
 */
 /*****************************************************************/
public native int Start(String strSerialNo);

/*****************************************************************/
/*
 * 函数名称：Stop
 * 输入参数：
 * 输出参数：
 * 返 回 值：
 * 功能描述：关闭程序
 */
 /*****************************************************************/
public native int Stop();

/*****************************************************************/
/*
 * 函数名称：GetLocalIp
 * 输入参数：
 * 输出参数：
 * 返 回 值：返回空表示获取错误
 * 功能描述：获取本机ip，调用Start之后才能获取成功
 */
 /*****************************************************************/
public native String GetLocalIp();
```

注意：要先调用 Start 再开启程序线程，在应用退出前一定要先调用 Stop，这样可以关闭所有已建立的连接并释放资源。

如何将接口引入的项目中：

1. 用 AndroidStudio 将你的项目打开；

2. 根据使用架构，四个库根据需要拷贝到项目 libs 文件夹下；

3. 将开发包里的 cloudclink.jar 也拷贝到 libs 根目录下在工程属性 ->Java Build Path->Libraries 中选择“Add External JARs”，选定 cloudclink.jar，确定后返回。

（也可以直接在 build.gradle 添加 compile files('libs/cloudclink.jar')）

3. 在项目的 build.gradle 中的 android 节点下增加：

```
    sourceSets {
        main {
            jniLibs.srcDirs = ['libs']
        }
}
```

4. 打开 AndroidManifest.xml 增加网络访问权限

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.cloudclink">
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

示例代码：

```
public class MainActivity extends AppCompatActivity {
    CloudClink cloude = null;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView result = (TextView) findViewById(R.id.result);
        Button btnStop = (Button) findViewById(R.id.button);
        cloude = new CloudClink();
        //从管理界面获取的key
        String key = "...";
        //启动客户端安全接入组件(只能启动一次，哪怕是已调用了Stop()也无法再重启,在应用启动后只能调用一次)，返回0表示成功，其它的为失败。
        //0：成功、1：代表验证串解析失败、2：序列号错误或过期（序列号验证）、3：已存在相同的终端（互斥）、4：其他错误
        int ret = cloude.Start(key);
        result.setText("result=" + ret);

        btnStop.setOnClickListener((v) -> {
                //注意：只有在退出应用时才调用stop()
                if (cloude != null) cloude.Stop();
                System.exit( status: 0);
        });
    }
    @Override
    // 退出调用Stop
    protected void onDestroy() {
        if (cloude != null)
            cloude.Stop();
        super.onDestroy();
    }
}
```

注意：

部分程序需要进行以下操作，否则会造成 Debug 版本可用，release 版本不可用

```
release {
/// ======== 关闭一些配置 ========
minifyEnabled false
shrinkResources false
/// ==========================================
}
```

5.4.3. iOS 接入

介绍通过 iOS 接入游戏盾 SDK 的操作方法

前提条件：

1、完成目标产品的转发规则配置。

2、从游戏盾 SDK 管理页面下载 SDK 安装包，并获取 iOS 版本的 SDK 包。

3、从游戏盾 SDK 管理列表获取目标产品的 SDK 密钥。

文件说明：

文件名	描述
yjs_sdk\lib\ios\libcloudclink_arm.a	只包含真机运行库
yjs_sdk\lib\ios\libcloudclink.a	包含真机和模拟器运行库
yjs_sdk\lib\ios\cloudclink.h	接口头文件

接口说明：

```
/*****************************************************************/
/*
 * 函数名称：ClinkStart
 * 输入参数：sdk配置密钥
 * 输出参数：无
 * 返 回 值：0:成功、1: key错误、2.key过期、3.已存在相同调用（调用成功）、4.其他错误、5.无节点可用
 * 功能描述：开启程序
 */
 /*****************************************************************/
int ClinkStart(const char * key);

/*****************************************************************/
/*
 * 函数名称：ClinkStop
 * 输入参数：
 * 输出参数：
 * 返 回 值：
 * 功能描述：关闭程序
 */
 /*****************************************************************/
void ClinkStop();

/*****************************************************************/
/*
 * 函数名称：GetLocalIp
 * 输入参数：
 * 输出参数：
 * 返 回 值：返回空表示获取错误
 * 功能描述：获取本机ip，调用ClinkStart之后才能获取成功
 */
 /*****************************************************************/
const char* GetLocalIp();
```

注意：要先调用 ClinkStart 再开启程序线程，在应用退出前一定要先调用 ClinkStop，这样可以关闭所有已建立的连接并释放资源。

发布项目是需要将项目的 Build Settings 中的 Enable Bitcode 设为 No。

如何将接口引入的项目中：

1. 用 Xcode 将你的项目打开；

2. 把 libcloudclink.a 的文件直接拖入你的项目中；

3. 把 cloudclink.h 文件拷贝到项目的目录下；

4. 把要调用接口的.m 文件改成扩展名的.mm，否则无法通过编译，因为如果文件是.m 编译器会以 c 的形式调用而不是 c++形式。

示例代码：

```
#import "ViewController.h"
#import "cloudclink.h" //包含头文件
@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];

    UITextField *txtMsg = [[UITextField alloc] init];
    txtMsg.frame = CGRectMake(60, 80, 200, 30);
    txtMsg.placeholder = @"返回的状态";
    [self.view addSubview:txtMsg];

    NSLog(@"start");
    //启动客户端安全接入组件(只能启动一次，哪怕是已调用了api.stop()也无法再重启,在应用启动后只能调用一次)，返回0表示成功，其它的为失败。
    //0:成功、1.代表验证串失败、2.序列号错误或过期（序列号验证，有可能网络问题）、3.已存在相同的终端、4:其他错误
    int ret = ClinkStart("...");
    NSString *ocStr= [NSString stringWithFormat:@"返回:%d",ret];
    txtMsg.text=ocStr;
    NSLog(@"start ok");
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}


@end
```

<正文完>