

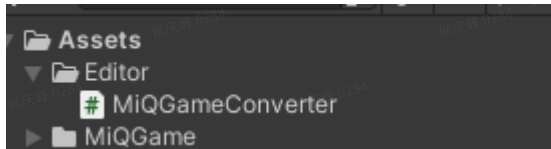
Unity转快游戏使用说明文档

转换工具链接

<http://tennews.cn/MiQGameConverter.unitypackage>

支持的unity版本**2021**、**2020**，**2019**。

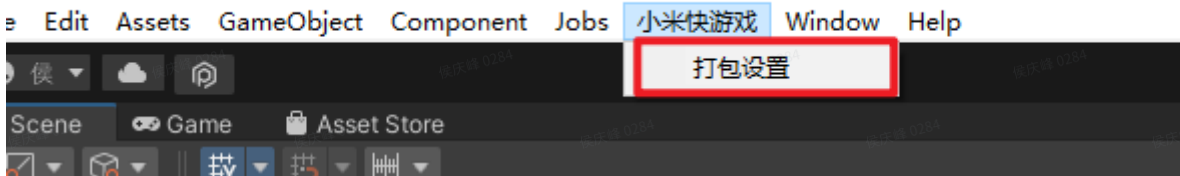
结构：



使用步骤

1. 点击菜单栏 小米快游戏/打包设置

MiQGameConverter - ConverterDemo - WebGL - Unity 2021.3.20f1 Personal <DX11>



2. 弹出操作窗口

小米快游戏介绍

链接

转换工具适配Unity版本: 2021.3.20f1, 2020.3.47f1, 2019.4.40f1c1

设置

游戏名称 小米快游戏测试

包名 com.demo.ch.mini

小米快游测试包名: com.demo.ch.mini

版本号 1.0.1

版本Code 101

输出目录 D:/worksapce/unity/MiQGameConverter/ 选择

游戏屏幕方向

☐ 横屏 ☒ 竖屏

包类型

☒ Debug

Debug签名文件目录 D:/worksapce/unity/MiQGameConverter/ 选择

☐ Release

Release签名文件目录 选择

注意: 签名文件生成方式, 请查看小米快游戏介绍文档!

包体超限说明:

如果生成的包体过大, 请把Build/build.data.txt和Build/build.wasm.txt放在网络上加载

☒ 网络资源

build.data.txt地址(http) http://10.241.193.6/Build/build.data.txt

build.wasm.txt地址(http) http://10.241.193.6/Build/build.wasm.txt

打包说明

(1) 生成快游戏rpk, 需要nodejs, 请确定已安装!

安装NodeJs

(2) 包名请确定是从小米快游戏中心申请的包名, 否则游戏加载不出来!

(3) 请把Assets/MiQGame/image/icon.png, 图片替换成自己的游戏图标!

(4) 快游戏总包体最大20M (包含分包), 如果超过, 请把资源移到网络存储当中, 使用网络加载方式!

打包

日志

参数说明:

- 游戏名称: 小米快游戏名称。
- 包名: 在小米快游戏中心申请的包名, 一定要填写正确的包名, 否则会导致加载失败。
- 版本号: 版本号。
- 版本code: 版本code。

- e. 输出目录：打包输出目录，此目录每次打包都会清空，请注意选择合适的目录。
- f. 游戏方向：横屏或者竖屏。
- g. 包类型：debug或者release包。release包需要release签名文件，请设定release签名文件所在的目录。
- h. 包体超限：由于快游戏包体大小限定，unity游戏资源较多的情况下，可以使用此项，把打出来资源放在服务器上，使用远程资源加载。
- i. 打包：执行打包操作。

3. 打包完成会打开打包目录。如下所示：

软件 (D:) > worksapce > unity > MiQGameConverter > QGameBuild > MiQGameBuild > MiQGame >

名称	修改日期	类型	大小
build	2023/6/6 11:34	文件夹	
dist	2023/6/6 11:34	文件夹	
image	2023/6/6 11:34	文件夹	
node_modules	2023/6/6 11:34	文件夹	
sign	2023/6/6 11:34	文件夹	
src	2023/6/6 11:34	文件夹	
unity	2023/6/6 11:34	文件夹	
babel.config.js	2023/4/13 16:10	JavaScript 源文件	1 KB
build.framework.js	2023/6/6 11:34	JavaScript 源文件	384 KB
main.js	2023/5/26 15:34	JavaScript 源文件	11 KB
manifest.json	2023/6/6 11:34	nddfile	1 KB
package.json	2023/4/18 14:11	nddfile	1 KB
package-lock.json	2023/6/6 11:34	nddfile	212 KB

MQGame即为小米快游戏工程。

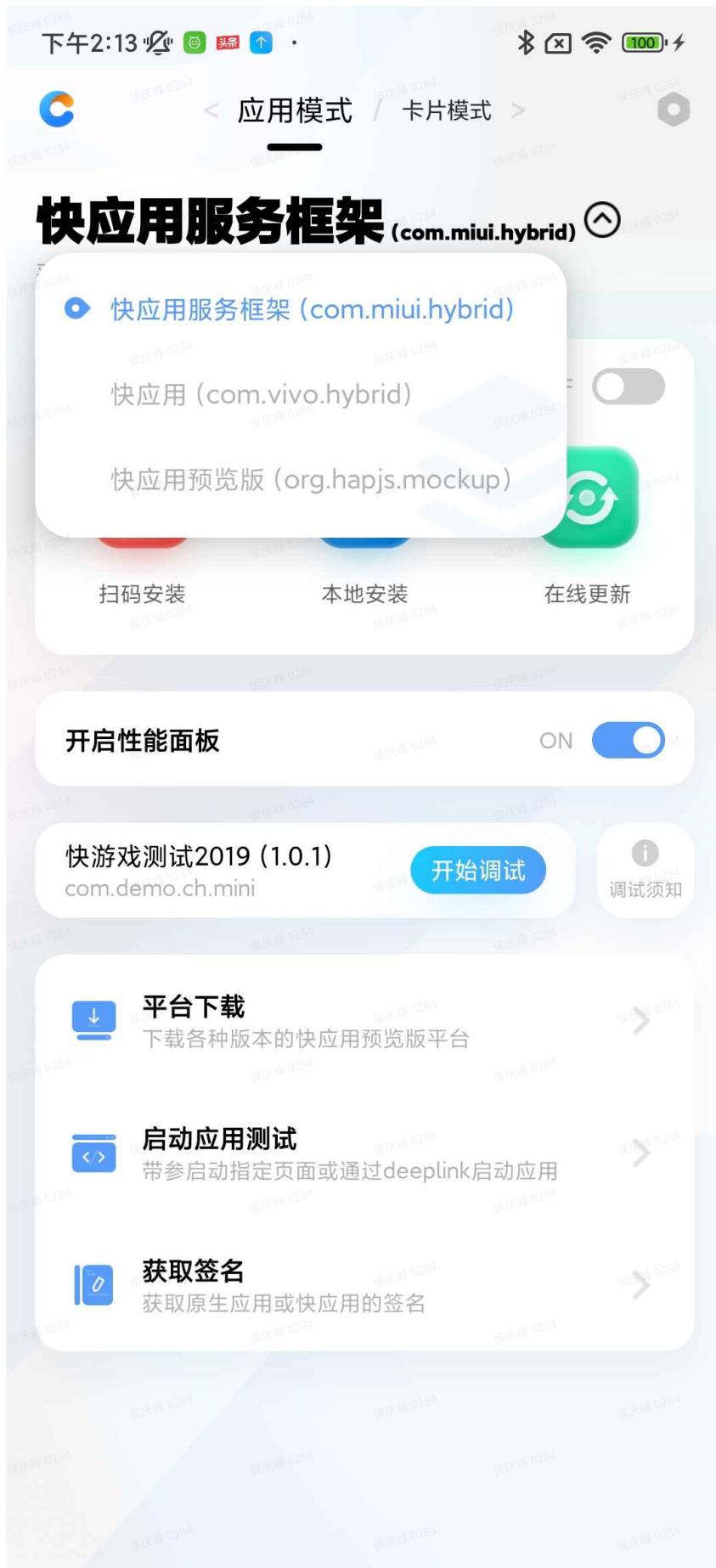
转换完成会在dist目录下生成快游包：

软件 (D:) > worksapce > unity > MiQGameConverter > QGameBuild > MiQGameBuild > MiQGame > dist

名称	修改日期	类型	大小
com.demo.ch.mini.debug.rpk	2023/6/6 11:34	RPK 文件	13,848 KB

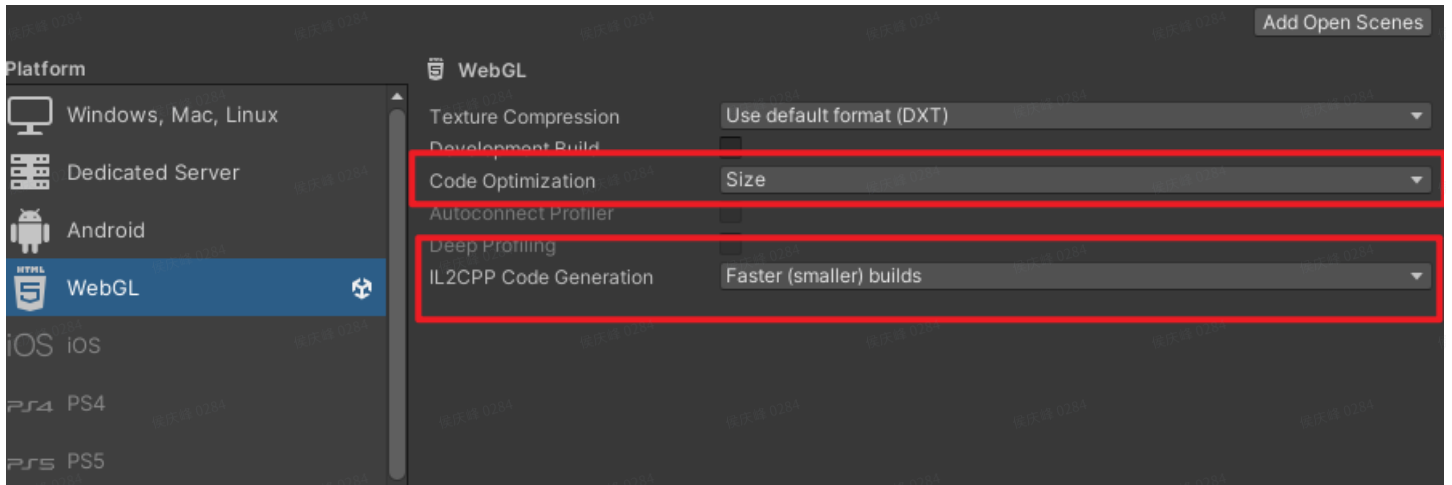
在此目录下，也可以手动执行npm命令生成快游戏包。

1. npm install命令安装快游戏依赖包；
2. npm run build 或者npm run release生成对应的debug包和release包。
3. npm run server，可以生成启动调试服务器，使用快游戏调试器（下载地址：<https://www.quickapp.cn/docCenter/post/69>），扫描二维码安装即可启动游戏。注意：需要把调试器环境切换为：com.miui.hybrid环境！！



优化建议

1. 包体过大，或者遇见out of memory，可以把这两两项改为如下选项，优化包体和内存。

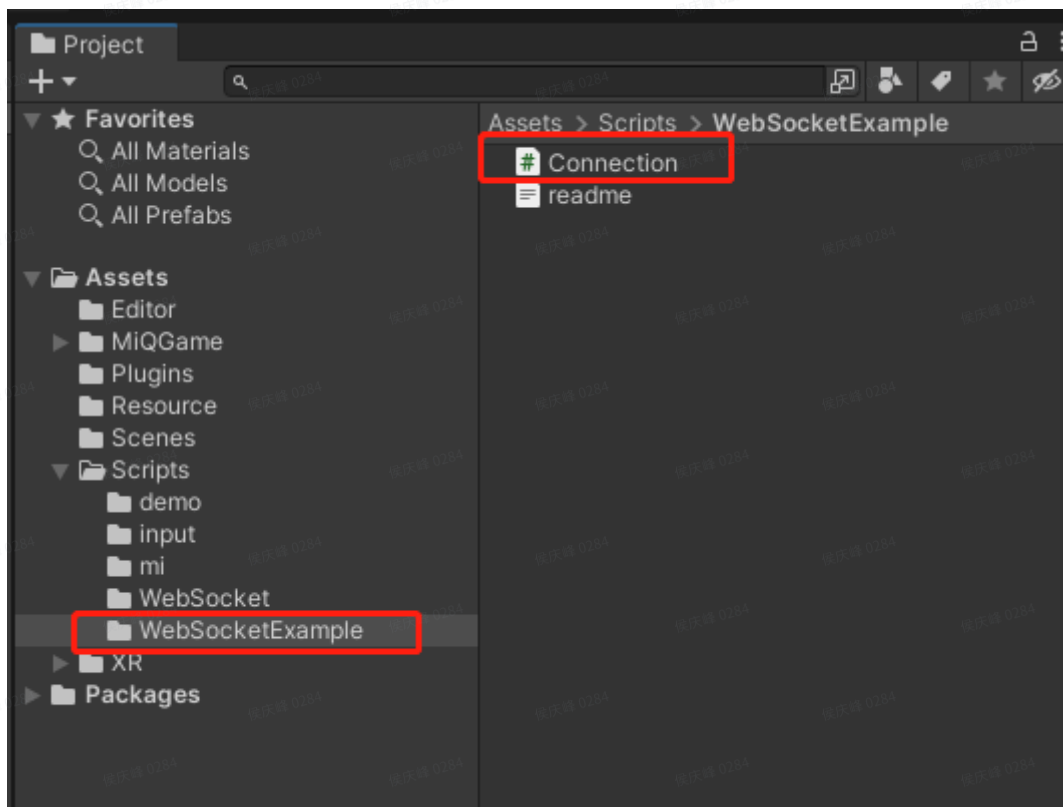


注意！！！Unity基础功能使用说明

unity转快游戏，有些unity功能是不能直接在快游戏里面使用的，需要在unity层使用的时候做一些适配工作。

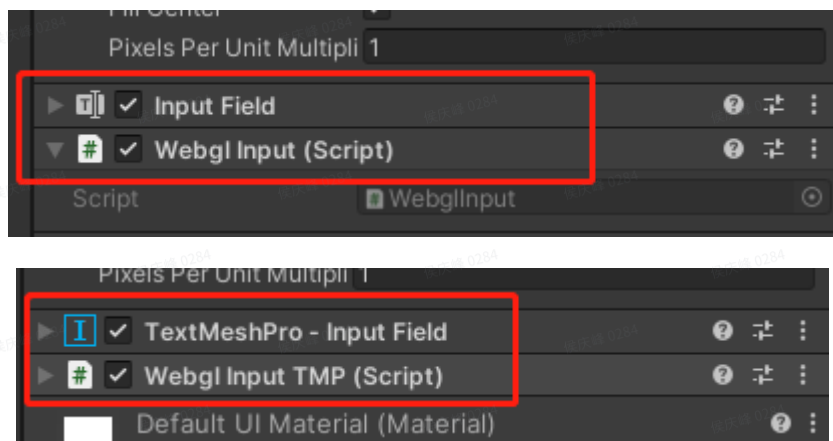
1. Websocket（可选）

示例：WebSocket服务器搭建完成后，在客户端场景物体上挂载示例脚本Connection，并修改Connection脚本中的IP地址即可。



2. InputField（必须）

在挂载输入框组件的物体上，需手动挂载一个脚本。如果使用的是 InputField (Legacy) 组件则手动挂载 WebglInput 脚本，使用的是 InputField (TMP)组件则手动挂载 WebglInputTMP 脚本。



快游戏功能接入

转换工具提供的快游戏功能有：登录、支付、广告、文件读写、创建游戏图标，key-value本地存储等功能。

1、初始化

初始化需要在调用sdk功能之前调用，确保初始化完成，才能使用SDK功能。

例子：

```
1 //初始化
2 MiBridge.Instance.Init();
```

2、登录和获取用户信息

```
1 //快游戏小米账号登录
2 MiBridge.Instance.Login((accountId, session) => {
3     MiBridge.Instance.QGLog("login account id={0}, session={1}", accountId, sess
4     MiBridge.Instance.GetUserInfo((nickName, avatarUrl, gender) =>
5     {
6         MiBridge.Instance.QGLog("get user info nickName={0}, avatar={1}, gender=
7     }, (code, msg) =>
8     {
9         MiBridge.Instance.QGLog("get user info error code={0}, msg={1}", code, m
10    });
11 }, (code, msg) => {
12     MiBridge.Instance.QGLog("login error code={0}, msg={1}", code, msg);
13 });
```

3、支付

支付接入需要服务器接入，具体服务器接入方式请查看小米开发者文档：

<https://dev.mi.com/distribute/doc/details?pId=1109>。

```
1 //订单信息
2 OrderInfo orderInfo = new OrderInfo
3 {
4     appId = "1", // 游戏唯一ID
5     appAccountId = 12, // 与登录接口返回的appAccountId一致
6     session = "1", // 与登录接口返回的session一致
7     cpOrderId = "1", // 游戏订单号
8     cpUserInfo = "", // cp透传信息（非空）
9     displayName = "1", // 支付的时候显示的商品名称
10    feeValue = 100, // 价格 单位分
11    sign = "1", // 签名 用于校验 具体生成方式，请查看：https://dev.mi
12 };
13
14 //支付接口
15 MiBridge.Instance.Pay(orderInfo, (success, code, msg) => {
16     MiBridge.Instance.QGLog("pay result success={0}, code={1}, msg={2}", success
17 });
```

4、广告

4.1、Banner广告

```
1 /// <summary>
2 /// 创建banner广告
3 /// </summary>
4 /// <param name="adId">广告id</param>
5 /// <param name="listener">监听器</param>
6 /// <returns></returns>
7 MiBridge.Instance.CreateBannerAd("81e6cbe35e56b53eebbc547fd1bc5614", new AdEven
8     onAdError = (code, msg) => {
9         MiBridge.Instance.QGLog("c# ad error");
10    },
11
12    onAdClose = (isEnd) => {
13        MiBridge.Instance.QGLog("c# ad close");
14    },
15
16    onAdLoad = (info) => {
17        MiBridge.Instance.QGLog("c# ad load");
18    },
19 });
```

```

20
21 /// <summary>
22 /// 隐藏广告
23 /// </summary>
24 /// <param name="adId"></param>
25 MiBridge.Instance.HideAd("81e6cbe35e56b53eebbc547fd1bc5614");
26
27 /// <summary>
28 /// 销毁广告，销毁后，如需重新显示，请先调用创建广告
29 /// </summary>
30 /// <param name="adId"></param>
31 MiBridge.Instance.DestroyAd("81e6cbe35e56b53eebbc547fd1bc5614");

```

4.2、插屏广告

```

1 /// <summary>
2 /// 创建插屏广告
3 /// </summary>
4 /// <param name="adId">广告id</param>
5 /// <param name="listener">监听器</param>
6 /// <returns></returns>
7 MiBridge.Instance.CreateInterstitialAd("f54f3dfc0ba63cf3dbf582816ee069d7", new A
8 {
9     onAdError = (code, msg) => {
10         MiBridge.Instance.QLog("c# insert ad error");
11     },
12
13     onAdClose = (end) => {
14         MiBridge.Instance.QLog("c# insert ad close");
15     },
16
17     onAdLoad = (info) => {
18         MiBridge.Instance.QLog("c# insert ad load");
19     },
20 });
21
22 /// <summary>
23 /// 展示插屏广告测试
24 /// </summary>
25 MiBridge.Instance.ShowAd("f54f3dfc0ba63cf3dbf582816ee069d7");
26
27 /// <summary>
28 /// 销毁插屏广告测试
29 /// </summary>
30 MiBridge.Instance.DestroyAd("f54f3dfc0ba63cf3dbf582816ee069d7");

```

4.3、Native广告

```
1  /// <summary>
2  /// 创建原生广告
3  /// </summary>
4  /// <param name="adId">广告id</param>
5  /// <param name="listener">监听器</param>
6  /// <returns></returns>
7  MiBridge.Instance.CreateNativeAd("da11b7e8c582ee7d1acf16a627ea6b34", new AdEvent
8  {
9      onAdError = (code, msg) => {
10         MiBridge.Instance.QGLog("c# native ad error");
11     },
12
13     onAdClose = (end) => {
14         MiBridge.Instance.QGLog("c# native ad close");
15     },
16
17     onAdLoad = (info) => {
18         MiBridge.Instance.QGLog("c# native ad load");
19     },
20 });
21
22  /// <summary>
23  /// 加载广告
24  /// </summary>
25  /// <param name="adId"></param>
26  MiBridge.Instance.LoadAd("da11b7e8c582ee7d1acf16a627ea6b34");
27
28  /// <summary>
29  /// 展示广告
30  /// </summary>
31  /// <param name="adId"></param>
32  MiBridge.Instance.ShowAd("da11b7e8c582ee7d1acf16a627ea6b34");
33
34  /// <summary>
35  /// 销毁广告，销毁后，如需重新显示，请先调用创建广告
36  /// </summary>
37  /// <param name="adId"></param>
38  MiBridge.Instance.DestroyAd("da11b7e8c582ee7d1acf16a627ea6b34");
```

4.4、激励视频广告

```

1  /// <summary>
2  /// 创建激励视频广告
3  /// </summary>
4  /// <param name="adId">广告id</param>
5  /// <param name="listener">监听器</param>
6  /// <returns></returns>
7  MiBridge.Instance.CreateRewardedVideoAd("77295ab0558fa54fc5cc9ed6a28b6da7", new
8  {
9      onAdError = (code, msg) => {
10         MiBridge.Instance.QGLog("c# reward ad error");
11     },
12
13     onAdClose = (end) => {
14         string info = "success";
15         if (!end) {
16             // end == true 说明激励视频播放完毕, 是正常结束
17             info = "failure";
18         }
19         MiBridge.Instance.QGLog($"c# reward ad close end={info}");
20     },
21
22     onAdLoad = (info) => {
23         MiBridge.Instance.QGLog("c# reward ad load");
24     },
25 });
26
27 /// <summary>
28 /// 加载广告
29 /// </summary>
30 /// <param name="adId"></param>
31 MiBridge.Instance.LoadAd("77295ab0558fa54fc5cc9ed6a28b6da7");
32
33 /// <summary>
34 /// 展示广告
35 /// </summary>
36 /// <param name="adId"></param>
37 MiBridge.Instance.ShowAd("77295ab0558fa54fc5cc9ed6a28b6da7");

```

4.5、互推盒子广告

```

1  /// <summary>
2  /// 展示互动盒子广告
3  /// 互动盒子广告介绍: https://dev.mi.com/distribute/doc/details?pId=1442
4  /// </summary>
5  /// <param name="adId">广告id</param>

```

```

6 /// <param name="type">
7 /// 广告类型:
8 /// 1. type=100, 互推盒子-九宫格
9 /// 2. type=120, 互推盒子-横幅
10 /// 3. type=130/140, 互推盒子-抽屉, 白色背景为130, 黑色背景为140
11 /// 4. type=150, 互推盒子-悬浮球
12 /// </param>
13 /// <param name="listener">监听器</param>
14 /// <returns></returns>
15 MiBridge.Instance.ShowRecommendAd("da11b7e8c582ee7d1acf16a627ea6b34", 100, (succ
16 {
17
18 });
19
20 /// <summary>
21 /// 关闭互动盒子广告
22 /// 互动盒子广告介绍: https://dev.mi.com/distribute/doc/details?pId=1442
23 /// </summary>
24 /// <param name="adId">广告id</param>
25 /// <param name="type">
26 /// 广告类型:
27 /// 1. type=100, 互推盒子-九宫格
28 /// 2. type=120, 互推盒子-横幅
29 /// 3. type=130/140, 互推盒子-抽屉, 白色背景为130, 黑色背景为140
30 /// 4. type=150, 互推盒子-悬浮球
31 /// </param>
32 /// <param name="listener">监听器</param>
33 MiBridge.Instance.CloseRecommendAd("da11b7e8c582ee7d1acf16a627ea6b34", 100, (suc
34 {
35
36 });

```

5、文件操作

5.1、读文件

```

1 MiBridge.Instance.ReadFile("log.txt", (data, len) =>
2 {
3     string test = System.Text.Encoding.ASCII.GetString(data);
4     MiBridge.Instance.QLog("MiBridge.Instance.ReadFile success {0}, {1}", len,
5 }, (msg) =>
6 {
7     MiBridge.Instance.QLog("read file failure {0}", msg);
8 });

```

5.2、写文件

```
1 string myString = "Hello world!!!";
2 byte[] data = Encoding.UTF8.GetBytes(myString);
3 MiBridge.Instance.WriteFile("log.txt", data, false, (success, msg) =>
4 {
5     MiBridge.Instance.QLog("write file {0}, {1}", success, msg);
6 });
```

5.3、删除文件

```
1 MiBridge.Instance.DeleteFile("log.txt", (success) => {
2 });
```

6、游戏图标

```
1 MiBridge.Instance.HasShortcut( has => {
2     if (has)
3     {
4         MiBridge.Instance.QLog("has short cut");
5     }
6     else
7     {
8         MiBridge.Instance.QLog("has not short cut");
9         MiBridge.Instance.CreateShortcut("便于打开游戏", (success, code, msg)
10             MiBridge.Instance.QLog("create short cut {0}, {1}, {2}", succes
11         });
12     }
13 });
```

7、KV存储

支持四种类型存储：int, float, double和string类型。

存和读操作：

```
1 MiBridge.Instance.SetKVInt("int", 1);
2 var kvInt = MiBridge.Instance.GetKVInt("int");
3 MiBridge.Instance.QLog("get int kv={0}", kvInt);
4
```

```
5 MiBridge.Instance.SetKVFloat("float", 2.0522222f);
6 var kvFloat = MiBridge.Instance.GetKVFloat("float");
7 MiBridge.Instance.QGLog("get float kv={0}", kvFloat);
8
9 MiBridge.Instance.SetKVDouble("dobule", 3.2656453453d);
10 var kvDouble = MiBridge.Instance.GetKVDouble("dobule");
11 MiBridge.Instance.QGLog("get dobule={0}", kvDouble);
12
13 MiBridge.Instance.SetKVString("string", " fsdfsdf ");
14 var kvString = MiBridge.Instance.GetKVString("string");
15 MiBridge.Instance.QGLog("get string={0}", kvString);
```

删除操作：

```
1 MiBridge.Instance.DeleteKV("int");
```

8、日志打印到快游戏平台

```
1 MiBridge.Instance.QGLog("hello world! ");
```