TPG 图片加速

SDK 接入

产品文档

腾讯云

SDK 接入

最近更新时间：2020-10-28 16:45:02

本文档以 Android 客户端接入 SDK 为例，介绍如何使用 SDK。

具体实现可以参考 Android 客户端 SDK Java 目录下的 TPGDemo 范例。

Android SDK 接入

步骤一：加载解码库

将解码库放在工程的 lib 目录下，工程代码里需加载 TPG 解码库：

```
System.loadLibrary ( “TPGDecoder” )
```

该解码库提供了以下 JNI 函数：

```
// interface using bitstream
private native int CreateDecoder(byte[] pStream);

private native int DecodeImage(long hObj, byte[] pStream, int index,
        TPGOutFrame tpgOutFrame);

private native int DecodeImageToBitmap(long hObj, byte[] pStream, int index,
        Bitmap bitmap, Integer delayTime);

private native void CloseDecoder(long hObj);

private native int ParseHeader(byte[] pStream, TPGFeature info);

private native int GetDelayTime(long hObj, byte[] pStream, int index);

private native byte[] GetAdditionalInfo(long hObj, byte[] pStream, int id);

// interface using file path as input
private native long CreateDecoder2(String tpg_path);

private native int DecodeImage2(long hObj, int index, TPGOutFrame tpgOutFrame);

private native int DecodeImageToBitmap2(long hObj, int index, Bitmap bitmap,
```

```
        Integer delayTime);

private native void CloseDecoder2(long hObj);

private native int ParseHeader2(String tpgpath, TPGFeature info);

private native int GetDelayTime2(long hObj, int index);

private native byte[] GetAdditionalInfo2(long hObj, int id);

private native int GetVersion(TPGVersionNum hObj);
```

步骤二：调用 TPGDecoder 接口

详细接口信息，见 API 接口 。

接口使用方法

1. 创建 TPGDecoder 对象，调用如下接口。确认输入图片是否为 tpg 格式。

```
private native int ParseHeader(byte[] pStream, TPGFeature info)
```

从传入的 `TPGFeature` 对象可以判断当前图片的宽度、高度、帧数和图片类型：

```
public class TPGFeature {
    int width;
    int height;
    int headerSize;
    int imageMode;
    int frameCount;
    int version;
}
```

图片类型可分为4类：

```
public static final int IMAGE_MODE_Normal = 0;
public static final int IMAGE_MODE_EncodeAlpha = 1;
public static final int IMAGE_MODE_BlendAlpha = 2;
public static final int IMAGE_MODE_Animation = 3;
```

```
public static final int IMAGE_MODE_AnimationWithAlpha = 4;
```

2. 图片为动图和非动图时实现方式如下：

- 如果图片为动图 `IMAGE_MODE_Animation` 或 `IMAGE_MODE_AnimationWithAlpha` ，可参考 TPGDemo 里 TPGDecoder.java 中的 `decodeOneFrame2 ()` 函数的实现。

```
public int decodeOneFrame2(int index, int[] outData, Bitmap bm,
        int[] delayTime) {

    int res = 0;
TPGOutFrame tpgOutFrame = new TPGOutFrame();
tpgOutFrame.pOutBuf = outData;
tpgOutFrame.dstWidth = mFeatureInfo.width;
tpgOutFrame.dstHeight = mFeatureInfo.height;
tpgOutFrame.fmt = Utils.FORMAT_BGRA;
if (DecodeImage2(mhDec, index, tpgOutFrame) > 0) {
    System.out.println("decode error: ");
}
delayTime[0] = tpgOutFrame.delayTime;

    bm.setPixels(outData, 0, mFeatureInfo.width, 0, 0, mFeatureInfo.width,
        mFeatureInfo.height);

    return res;
}
```

- 如果图片为非动图，可参考 TPGDemo 里 TPGDecoder.java 中的 `decodeTPG2()` 函数的实现。

```
public Bitmap decodeTPG2(String tpgPath, int format, int dstWidth) {
    Bitmap bm = null;

    int dstHeight = 0;

    TPGFeature info = new TPGFeature();

    int res = ParseHeader2(tpgPath, info);

    if (res != Utils.TPG_STATUS_OK) {
    return null;
```

```
}
mhDec = CreateDecoder2(tpgPath);

    if (mhDec == 0) {
    return null;
}

    imageWidth = info.width;
imageHeight = info.height;

    dstHeight = (int) ((double) imageHeight / (double) imageWidth * dstWidth);

    if (dstWidth > imageWidth || dstHeight > imageHeight) {
    dstWidth = imageWidth;
    dstHeight = imageHeight;
}
// System.out.println("createBitmap start!" + imageWidth*imageHeight*4);
bm = Bitmap.createBitmap(dstWidth, dstHeight, Bitmap.Config.ARGB_8888);
if(info.imageMode==Utils.IMAGE_MODE_Normal)
{
    bm.setHasAlpha(false);
}

    // System.out.println("createBitmap end!");
if (null == bm) {
    System.out.println("no memory!");
}
int delayTime = 0;
if (DecodeImageToBitmap2(mhDec, 0, bm, delayTime) > 0) {
    System.out.println("decode image error!");
}

    //TPGVersionNum foo = new TPGVersionNum();

    //GetVersion(foo);
//System.out.println("version num:"+foo.num+"version str:"+foo.str);
CloseDecoder2(mhDec);
mhDec = 0;

    return bm;
```

```
}
```