FaceID Lite 集成文档（Android 版）

目录

1 简介

FaceID Lite 是针对移动端网页身份验证服务，活体认证，主要应用于微信端和 App 原生+H5 的开发上。下面介绍在 Android 端 App 开发中应用 FaceID Lite 的认证服务。

1.1 技术指标

	Android	iOS
硬件指标	拥有前置摄像头，麦克风	iPhone4s 及以上
系统指标	Android 4.0 及以上	iOS6.0 及以上
软件指标	无	无
网络要求	推荐使用 3G/4G 网络，或连接 WIFI	

注：移动端身份验证服务要求手机浏览器应有摄像头相机、录音等相关权限，并支持 HTML5 技术。

1.2 产品规格

移动端身份验证服务（ FaceID Lite ） 产品提供了一个在手机端，通过网页完成实名验证的功能。验证流程分为三步，采集身份证信息、活体验证、人脸比对。

步骤	功能	功能说明	功能选择	返回结果
1	身份证、姓名的获取	获取实名验证人员的身份证信息。	传入身份证号、姓名	身份证号 姓名
			手动输入 身份证号、姓名	身份证号 姓名
			拍摄 身份证正面	身份证号 姓名 身份证正面识别

				结果、五分类信息 身份证正面图 用户相关操作记录
			拍摄 身份证正面、背面	身份证号 姓名 身份证正面识别结果、五分类信息 身份证正面图 身份证背面识别结果、五分类信息 身份证背面图 用户相关操作记录
2	朗读数字活体验证	通过技术手段验证进行实名验证的为真人， 并采集验证人人像信息。	录制视频通过活体验证	活体成功后质量最佳的图 用户相关操作记录
3	人脸验证服务	通过比对验证人人像与身份证图像、数据源图像、 其他可信图像完成对验证人的实名验证。 做到真正的人证合一 验证。	FaceID Verify 功能 是否用数据源 是否有 image_ref 是否有身份证人脸图	活体人脸与数据源比对结果* 活体人脸与身份证比对结果* 活体人脸与 image_ref 的比对结果* 活体人脸假脸判断 身份证号和姓名的可信任度判断

2 接口介绍

详细 API 请求的参数字段以及返回的信息请参考详细请参考 FaceID 官网文档中心。

2.1 GetToken

URL

POST https://api.megvii.com/faceid/lite/get_token

说明

此接口提供 FaceID Lite 验证服务，通过此接口客户可以获得一个用于实名验证的 token（token 唯一且只能使用一次）。接口同时还能帮助完成人脸比对，并在完成验证后自动将人脸比对结果返回，方便集成开发。

2.2 DoVerification

URL

GET https://api.megvii.com/faceid/lite/do

说明

此接口提供 FaceID Lite 验证服务，通过 GetToken 接口获得 token，可利用此接口转跳到对应的网页验证。

2.3 GetResult

URL
GET https://api.megvii.com/faceid/lite/get_result

说明

此接口提供活体结果反查功能，可以以 biz_id 为索引对 FaceID Lite 验证结果进行反查。

3 集成

首先通过 Lite-GetToken 获取 token，然后通过在 Android 原生控件 WebView 中加载 Lite-DoVerification 网址去实现 Android 在原生 App 调用移动端 HTML5 界面，技术点：主要是在 webview 调用 H5 界面中实现原生调用相机拍照和录像的功能，下面是集成中可能需要注意的地方。

3.1 权限

首先在配置清单文件需要配置的权限：

```
<uses-permission android:name="android.permission.READ_INTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.CAMERA" />
```

上面权限配置在 Android4.0 以上，Android6.0 及以下是可以实现的，但是在部分 Android6.0 的手机中由于系统深度定制的原因（例如华为 P9），必须唤起用户去开启用户权限才可以正常实现，需要在调用的 Activity 中实现下列代码。

```
private static String[] PERMISSIONS_STORAGE = {
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.WRITE_EXTERNAL_STORAGE,
        Manifest.permission.CAMERA
};

//persmission method.
private  void getPermissions(Activity activity) {
    // Check if we have read or write permission
    int writePermission = ActivityCompat.checkSelfPermission(activity,
            Manifest.permission.WRITE_EXTERNAL_STORAGE);
    int readPermission = ActivityCompat.checkSelfPermission(activity,
            Manifest.permission.READ_EXTERNAL_STORAGE);
    int cameraPermission = ActivityCompat.checkSelfPermission(activity,
            Manifest.permission.CAMERA);
    if (writePermission != PackageManager.PERMISSION_GRANTED
            || readPermission != PackageManager.PERMISSION_GRANTED
            ||cameraPermission!=PackageManager.PERMISSION_GRANTED) {
        // We don't have permission so prompt the user
        ActivityCompat.requestPermissions(
                activity,
                PERMISSIONS_STORAGE,
                REQUEST_EXTERNAL_STORAGE

        );
    }
```

3.2 配置

```
webSettings.setJavaScriptEnabled(true);// 支持JS
webSettings.setLoadWithOverviewMode(true);
webSettings.setBuiltInZoomControls(true);
webSettings.setPluginState(WebSettings.PluginState.ON);
webSettings.setAllowFileAccess(true);
webSettings.setUseWideViewPort(true);
webSettings.setLoadWithOverviewMode(true);
webSettings.setSupportZoom(true);
if (Build.VERSION.SDK_INT > Build.VERSION_CODES.HONEYCOMB) {
    // Hide the zoom controls for HONEYCOMB+
    webSettings.setDisplayZoomControls(false);
}
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
    WebView.setWebContentsDebuggingEnabled(true);
}
```

设置 webviewclient 去调用网页

```
webview.setWebViewClient(new WebViewClient() {
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url) {
        view.loadUrl(url);
        return true;
    }
});
webview.loadUrl(url);
}
```

3.3 实现回调

用 Android 原生去实现 H5 界面对相机拍照和录像功能，代码需要实现对

1WebChromeClient 的调用。

```
webview.setWebChromeClient(new WebChromeClient() {
```

在 Android5.0 以下的方法重写 openFileChooser（ ）

```
//for  Android 4.0+
public void openFileChooser(ValueCallback<Uri> uploadMsg, String acceptType, String capture) {
```

在 Android5.0 以上的调用重写 onShowFileChooser（ ）

```
@SuppressLint("NewApi")
@Override
public boolean onShowFileChooser(WebView webView, ValueCallback<Uri[]> filePathCallback,
                                 WebChromeClient.FileChooserParams fileChooserParams) {
```

通过判断回调函数的 acceptType 参数的内容是 image/*还是 video/*进而选择调用拍照还是录像，需要注意的是 Android 7.0 之后需要使用 FileProvider 兼容拍照，需要配置以下参数：

1）AndroidManifest.xml 文件中需要声明 FileProvider，红框位置为当前 App 的包名。

```
<provider
    android:name="android.support.v4.content.FileProvider"
    android:authorities="com.megvii.lite.fileprovider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/file_paths"/>
</provider>
```

2）编写 resource xml file

```
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path name="my_images" path="Android/data/com.megvii.lite/files/Pictures/" />
    <external-path name="images" path="Pictures/" />
    <external-path name="dcim" path="DCIM/" />
</paths>
```

详细实现代码如下：

```
package com.megvii.lite;
import android.Manifest;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.support.annotation.RequiresApi;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v4.content.FileProvider;
```

```
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.webkit.ConsoleMessage;
import android.webkit.ValueCallback;
import android.webkit.WebChromeClient;
import android.webkit.WebResourceRequest;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import java.io.File;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
import static android.os.Build.VERSION_CODES.M;

/**
 * Created by zjh
 */
public class WebviewActivity extends Activity {
    private WebView webview;
    private String url;
    private ValueCallback<Uri[]> mFilePathCallback;
    private ValueCallback<Uri> nFilePathCallback;
    private String mCameraPhotoPath;
    public static final int INPUT_FILE_REQUEST_CODE = 1;
    public static final int INPUT_VIDEO_CODE = 2;
    private static final int REQUEST_CAMERA_CODE = 1;
    private Uri photoURI;
    private void getPermissions(Activity activity) {
        if (android.os.Build.VERSION.SDK_INT >= M) {
            if (ContextCompat.checkSelfPermission(this,
                    Manifest.permission.CAMERA)
                    != PackageManager.PERMISSION_GRANTED) {
                //进行权限请求
                ActivityCompat.requestPermissions(this,
                        new String[]{Manifest.permission.CAMERA},
                        REQUEST_CAMERA_CODE);
            }
        }
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_webview);
        getPermissions(this);
        initData();

    }
    private void initData() {
        String data = getIntent().getStringExtra("data");
        url = "https://api.megvii.com/faceid/lite/do?token=" + data;
        webview = (WebView) findViewById(R.id.webview);
        WebSettings webSettings = webview.getSettings();
        webview.setScrollBarStyle(
        WebView.SCROLLBARS_OUTSIDE_OVERLAY);
        webview.setScrollbarFadingEnabled(false);
        webSettings.setJavaScriptEnabled(true);
        webSettings.setLoadWithOverviewMode(true);
        webSettings.setBuiltInZoomControls(true);
        webSettings.setPluginState(WebSettings.PluginState.ON);
        webSettings.setAllowFileAccess(true);
        webSettings.setUseWideViewPort(true);
        webSettings.setLoadWithOverviewMode(true);
        webSettings.setSupportZoom(true);
        if (Build.VERSION.SDK_INT >
Build.VERSION_CODES.HONEYCOMB) {
            // Hide the zoom controls for HONEYCOMB+
            webSettings.setDisplayZoomControls(false);
        }
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
            WebView.setWebContentsDebuggingEnabled(true);
        }
        webview.loadUrl(url);
        webview.setWebViewClient(new WebViewClient() {
            @Override
            public boolean shouldOverrideUrlLoading(WebView view, String url)
{
                 view.loadUrl(url);
                return true;
            }

            @Override
            public void onPageStarted(WebView view, String url, Bitmap favicon)
{

                super.onPageStarted(view, url, favicon);
```

```
            }

            @Override
            public void onPageFinished(WebView view, String url) {
                super.onPageFinished(view, url);

            }

            @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
            @Override
            public boolean shouldOverrideUrlLoading(WebView view,
WebResourceRequest request) {
                String url = request.getUrl().toString();
                view.loadUrl(url);
                return true;
            }

            @Override
            public void onLoadResource(WebView view, String url) {
                super.onLoadResource(view, url);
            }


        });
        webview.setWebChromeClient(new WebChromeClient() {
            @Override
            public void onProgressChanged(WebView view, int newProgress) {
                super.onProgressChanged(view, newProgress);
            }

            @Override
            public void onReceivedTitle(WebView view, String title) {
                super.onReceivedTitle(view, title);
            }

            //for    Android 4.0+
            public void openFileChooser(ValueCallback<Uri> uploadMsg, String
acceptType, String capture) {
                if (nFilePathCallback != null) {
                    nFilePathCallback.onReceiveValue(null);
                }
                nFilePathCallback = uploadMsg;
                if ("image/*".equals(acceptType)) {
                    Intent takePictureIntent = new
```

```
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                        if
(takePictureIntent.resolveActivity(getPackageManager()) != null) {
                            File photoFile = null;
                            try {
                                photoFile = createImageFile();
                                takePictureIntent.putExtra("PhotoPath",
mCameraPhotoPath);
                            } catch (IOException ex) {
                                Log.e("TAG", "Unable to create Image File", ex);
                            }
                            if (photoFile != null) {
                                mCameraPhotoPath = "file:" +
photoFile.getAbsolutePath();

takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT,
                                        Uri.fromFile(photoFile));
                            } else {
                                takePictureIntent = null;
                            }
                        }
                        startActivityForResult(takePictureIntent,
INPUT_FILE_REQUEST_CODE);
                    } else if ("video/*".equals(acceptType)) {
                        Intent takeVideoIntent = new
Intent(MediaStore.ACTION_VIDEO_CAPTURE);
                        if
(takeVideoIntent.resolveActivity(getPackageManager()) != null) {
                            startActivityForResult(takeVideoIntent,
INPUT_VIDEO_CODE);
                        }
                    }
                }
                @SuppressLint("NewApi")
                @Override
                public boolean onShowFileChooser(WebView webView,
ValueCallback<Uri[]> filePathCallback,

WebChromeClient.FileChooserParams fileChooserParams) {
                    if (mFilePathCallback != null) {
                        mFilePathCallback.onReceiveValue(null);
                    }
                    mFilePathCallback = filePathCallback;
                    String[] acceptTypes = fileChooserParams.getAcceptTypes();
```

```
                    if (acceptTypes[0].equals("image/*")) {
                        Intent takePictureIntent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                        if
(takePictureIntent.resolveActivity(getPackageManager()) != null) {
                            File photoFile = null;
                            try {
                                photoFile = createImageFile();
                                takePictureIntent.putExtra("PhotoPath",
mCameraPhotoPath);
                            } catch (IOException ex) {
                                Log.e("TAG", "Unable to create Image File", ex);
                            }
                            //适配 7.0
                            if(Build.VERSION.SDK_INT > M) {
                                if (photoFile != null) {
                                    photoURI =
FileProvider.getUriForFile(WebviewActivity.this,

BuildConfig.APPLICATION_ID+".fileprovider", photoFile);

takePictureIntent.addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);

takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, photoURI);
                                }
                            }else{
                                if (photoFile != null) {
                                    mCameraPhotoPath = "file:" +
photoFile.getAbsolutePath();

takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT,
                                            Uri.fromFile(photoFile));
                                } else {
                                    takePictureIntent = null;
                                }
                            }
                        }
                        startActivityForResult(takePictureIntent,
INPUT_FILE_REQUEST_CODE);
                    } else if (acceptTypes[0].equals("video/*")) {
                        Intent takeVideoIntent = new
Intent(MediaStore.ACTION_VIDEO_CAPTURE);
                        if
(takeVideoIntent.resolveActivity(getPackageManager()) != null) {
```

```
                                startActivityForResult(takeVideoIntent,
INPUT_VIDEO_CODE);
                            }
                        }
                        return true;
                    }

                    @Override
                    public boolean onConsoleMessage(ConsoleMessage consoleMessage)
{
                        return true;
                    }
                });

                webview.loadUrl(url);

        }

        private File createImageFile() throws IOException {
                String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss",
Locale.CHINA).format(new Date());
                String imageFileName = "JPEG_" + timeStamp + "_";
                File storageDir =
getExternalFilesDir(Environment.DIRECTORY_PICTURES);
                File image = File.createTempFile(
                        imageFileName,  /* 前缀 */
                        ".jpg",         /* 后缀 */
                        storageDir      /* 文件夹 */
                );
                 mCameraPhotoPath = image.getAbsolutePath();
                return image;
        }

        @Override
        public void onActivityResult(int requestCode, int resultCode, Intent data) {
                if (requestCode != INPUT_FILE_REQUEST_CODE && requestCode !=
INPUT_VIDEO_CODE) {
                    super.onActivityResult(requestCode, resultCode, data);
                    return;
                }
                Uri[] results = null;
                Uri mUri = null;
                if (resultCode == Activity.RESULT_OK && requestCode ==
INPUT_FILE_REQUEST_CODE) {
```

```
                if (data == null) {
                    if(Build.VERSION.SDK_INT > M) {
                        mUri=photoURI;
                        results=new Uri[]{mUri};
                    }else{
                        if (mCameraPhotoPath != null) {
                            mUri = Uri.parse(mCameraPhotoPath);
                            results = new Uri[]{Uri.parse(mCameraPhotoPath)};
                        }
                    }
                } else {
                    Uri nUri = data.getData();
                    if (nUri != null) {
                        mUri =nUri;
                        results = new Uri[]{nUri};
                    }
                }
            } else if (resultCode == Activity.RESULT_OK && requestCode ==
INPUT_VIDEO_CODE) {
                mUri = data.getData();
                results = new Uri[]{mUri};
            }
            if (Build.VERSION.SDK_INT < Build.VERSION_CODES.LOLLIPOP) {
                nFilePathCallback.onReceiveValue(mUri);
                nFilePathCallback = null;
                return;
            } else {
                mFilePathCallback.onReceiveValue(results);
                mFilePathCallback = null;
                return;
            }
        }

        @Override
        public boolean onKeyDown(int keyCode, KeyEvent event) {
            if ((keyCode == KeyEvent.KEYCODE_BACK) &&
webview.canGoBack()) {
                webview.goBack();
                return true;
            }
            return super.onKeyDown(keyCode, event);
        }

}
```