|  |  |
| --- | --- |
|  | **iOnePass**  **for Android**  **개발 가이드** |

**IDEATEC Co, Ltd.**

목 차

[1 개 요 3](#_Toc41825081)

[1.1 소 개 3](#_Toc41825082)

[1.2 참고사항 3](#_Toc41825083)

[1.3 개발지원 3](#_Toc41825084)

[2 구 성 3](#_Toc41825085)

[2.1 시스템구성 및 흐름도 3](#_Toc41825086)

[3 프로젝트 환경설정 3](#_Toc41825087)

[3.1 라이브러리 구성 3](#_Toc41825088)

[3.2 프로젝트 설정 4](#_Toc41825089)

[3.2.1 Android Studio에서 프로젝트 설정. 4](#_Toc41825090)

[3.2.2 보안 네트워크 통신 정책 변경 4](#_Toc41825091)

[4 iOnePassLib 개발 가이드 4](#_Toc41825092)

[4.1 호출 흐름도 4](#_Toc41825093)

[4.1.1 객체생성 5](#_Toc41825094)

[4.1.2 등록 단계 5](#_Toc41825095)

[4.1.3 인증 단계 6](#_Toc41825096)

[4.1.4 해제 단계 7](#_Toc41825097)

# **1 개 요**

## **1.1 소 개**

|  |
| --- |
| iOnePassLib은 고객이 이용하는 서비스에 대한 1차 로그인 이후에, 생체 인증을 이용한 2차 인증을 제공하는 종합 솔루션입니다. 비밀번호를 사용하지 않는 FIDO인증 표준기술 중 UAF1.1 (Universal Authentication Framework)를 이용하여, 지문, 얼굴 인식 등 생체 정보를 통해 인증과정을 수행하며, 이 전 과정을 고객사의 어플리케이션에 쉽게 적용하여 개발하도록 라이브러리 가이드를 제공합니다. |

## **1.2 참고사항**

|  |
| --- |
| - Android SDK Version 최소 설정 21이상  - Android Studio 3.6.3 기준 작성  - 지원 UAF 버전 : 1.1 |

## **1.3 개발지원**

|  |
| --- |
| 이데아텍 보안사업부 연락처  E-MAIL : hanit@ideatec.co.kr  TEL : 02-4060-3868 |

# **2 구 성**

## **2.1 시스템구성 및 흐름도**

|  |
| --- |
|  |

※ 고객사 앱에서 필요시 호출하면 인증 후 결과를 고객사 앱으로 (성공, 실패)여부를 전달합니다.

# **3 프로젝트 환경설정**

## **3.1 라이브러리 구성**

|  |
| --- |
| iOnePass\_AOS\_1.X.XX.yyyyMMddii  ├doc : 각종 가이드 문서가 있습니다.(본 문서 포함)  ┖library : Android용 라이브러리 패키지입니다. |

## **3.2 프로젝트 설정**

### 3.2.1 Android Studio에서 프로젝트 설정.

|  |
| --- |
| **iOnePassLib for Android SDK 설치하기**  1. 제공받은 iOnePassLib-1.x.x.zip 압축 파일에서 Android용 iOnePassLib SDK 압축 파일을 풉니다.  2. iOnePassLib.aar 파일을 [프로젝트 root폴더] \app\libs 폴더에 복사를 합니다.  3. Android Studio를 열고 작업할 프로젝트를 불러 옵니다. build.gradle (Module:app) 파일에 아래 내용을 추가합니다.  repositories **{** flatDir **{** dirs 'libs' **} }**  dependencies **{**  …  implementation 'kr.co.ideatec.ionepasslib:iOnePassLib:1.0@aar'  …    4. gson 라이브러리의 추가  iOnePass 라이브러리에서는 데이터 전달시 json형태로 전달하기 때문에 gson 라이브러리의 추가가 필요합니다. (build.gradle – app범위)  dependencies **{** implementation fileTree(dir: 'libs', include: ['\*.jar'])  implementation 'com.google.code.gson:gson:2.8.6'  …  5. Firebase Cloude Messaging (이하 FCM) 서비스사용을 위한 라이르버리 추가가 필요 합니다.  dependencies **{** implementation fileTree(dir: 'libs', include: ['\*.jar']) …   implementation 'com.google.firebase:firebase-analytics:17.4.2'  implementation 'com.google.firebase:firebase-messaging:20.2.0'  implementation 'com.google.firebase:firebase-core:17.4.2'  implementation 'com.google.firebase:firebase-auth:19.3.1'  … |

### 3.2.2 보안 네트워크 통신 정책 변경

일부 HTTP통신을 사용할 필요가 있는 경우에는 AndroidManifast.xml에 아래 내용을 추가 합니다.

|  |
| --- |
| <uses-library android:name="org.apache.http.legacy" android:required="false"/> |

※ 더 자세한 내용은 “iOnePassLib\_Android\_04.ERROR\_CODE\_정의서\_20200531\_Ver1.0” 을 참고하세요.

# **4 iOnePassLib 개발 가이드**

## **4.1 호출 흐름도**

|  |
| --- |
|  |

※ 2차인증 모듈 호출은 환경설정 뒤 결과 상태를 받을 Listener를 정의 및 설정, 결과를 수신하여 이후 작업을 하는 순서로 진행됩니다. 아래 단계별 예제 소스와 추가 설명이 기입 되어있습니다.

### 4.1.1 객체생성

|  |
| --- |
| import kr.co.ideatec.ionepasslib.IOnePassLib; import kr.co.ideatec.ionepasslib.listener.iOnePassLibNetworkListener;   public class RegisterActivity extends AppCompatActivity { …   IOnePassLib mIOPLib;  private static final String *TAG* = RegisterActivity.class.getSimpleName();   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_register*);  this.initializeView();  this.initEvent();   }   public void initializeView() {//객체생성  …  mIOPLib = IOnePassLib.getInstance();  }  … |

※ 상기코드는 샘플 앱에 구현한 것 중 일부만을 표시 한 것입니다.

### 4.1.2 등록 단계

#### *4.1.2.1 요청 방법*

|  |
| --- |
| …  mIOPLib.reqRegistration(RegisterActivity.this, username,new iOnePassLibNetworkListener() { // 등록작업시에 .reqResigstration을 호출한다.  @Override  public void didError(String errMessage) {  Toast.*makeText*(RegisterActivity.this.getApplicationContext(),"등록실패 : " + errMessage,Toast.*LENGTH\_SHORT*);  }   @Override  public void didSuccess() {  Toast.*makeText*(RegisterActivity.this.getApplicationContext(),"등록성공",Toast.*LENGTH\_SHORT*);  }  });  }  … |

※ 등록 단계시 필요한 메소드를 호출 합니다. mIOPLib.reqResistration(); 자세한 내역은 별첨으로 제공된 API Reference를 참고해 주세요.

#### *4.1.2.2 onActivityResult 처리*

|  |
| --- |
| …  @Override  protected void onActivityResult(int requestCode, int resultCode, Intent data) {  super.onActivityResult(requestCode, resultCode, data);  Toast.*makeText*(this, "REQUEST CODE : " + requestCode + " / RESULT CODE : " + resultCode, Toast.*LENGTH\_SHORT*).show();   mIOPLib.sendUAFRegResponseData(this, requestCode, resultCode, data,new iOnePassLibNetworkListener() {   @Override  public void didSuccess() {   Log.*d*(*TAG*, "IOP\_FIDO\_RP\_CLIENT.handleSuccess : ");  }   @Override  public void didError(String errorMessage) {  Log.*d*(*TAG*, "IOP\_FIDO\_RP\_CLIENT.handleError : " + errorMessage);  }  });  Toast.*makeText*(RegisterActivity.this.getApplicationContext(),"등록성공",Toast.*LENGTH\_SHORT*);  }  });  }  … |

※ 등록 단계시 필요한 메소드를 호출 합니다. mIOPLib.reqResistration(); 자세한 내역은 별첨으로 제공된 API Reference를 참고해 주세요.

### 4.1.3 인증 단계

#### *4.1.3.1 요청 방법*

인증 단계는, 등록 완료된 사용자 중 Push Message를 받은 사용자에 한하여 진행됨을 기준으로 합니다.

|  |
| --- |
| …  mIOPLib.reqAuth(mContext,username,transMessage, getTid(), new iOnePassLibNetworkListener() {  @Override  public void didError(String errMessage) {  Log.*i*(*TAG*, "인증실패 : " + errMessage);  failDialog();  }   @Override  public void didSuccess() {  Log.*i*(*TAG*, "인증성공");  } });  … |

※ 인증 단계시 필요한 메소드를 호출 합니다. mIOPLib.reqAuth(); 자세한 내역은 별첨으로 제공된 API Reference를 참고해 주세요.

※ http://192.168.219.140:8082/RPserver페이지 로그인시에 Push Message가 테스트 기기로 전달되며 인증 단계 테스트가 가능합니다. 단 테스트 로그인 페이지는 환경에 따라 ip는 변경될 수 있습니다.

#### *4.1.3.2 onActivityResult 처리*

|  |
| --- |
| …  @Override  protected void onActivityResult(int requestCode, int resultCode, Intent data) {  super.onActivityResult(requestCode, resultCode, data);  Toast.*makeText*(this, "REQUEST CODE : " + requestCode + " / RESULT CODE : " + resultCode, *LENGTH\_SHORT*).show();  Log.*i*(*TAG*, "REQUEST CODE : " + requestCode + " / RESULT CODE : " + resultCode);   //1001값으로 왔을때 respond 성공으로 봅니다. 이 사유로 주석처리 하였습니다. Hanit - 2020.05.29. 한인택 // if(requestCode != 1000){ // failDialog(); // return; // }   mIOPLib.sendUAFAuthResponseData(requestCode, resultCode, data, new IOnePassLibNetworkListener() {   @Override  public void didSuccess() {  Log.*i*(*TAG*,"onActivityResult >> sendUAFAuthResponseData >> Success ");  success();  }   @Override  public void didError(String errorMessage) {  Log.*i*(*TAG*,"onActivityResult >> sendUAFAuthResponseData >> error : " + errorMessage);  failDialog();  }  });  … |

### 4.1.4 해제 단계

#### *4.1.4.1 요청 방법*

|  |
| --- |
| …  mIOPLib.reqDeregistration(mContext, username, new IOnePassLibNetworkListener() {  @Override  public void didError(String errMessage) {  butils.hideProgressDialog();  butils.showDialogOneBtn(DeregisterActivity.this, "등록해제에 실패하였습니다. 잠시후 다시 요청해주세요", "확인", new DialogInterface.OnClickListener() {  @Override  public void onClick(DialogInterface dialog, int which) {  finish();  }  });  Log.*i*(*TAG*, "등록해제 실패 : " + errMessage);  }   @Override  public void didSuccess() {  butils.hideProgressDialog();  butils.showDialogOneBtn(DeregisterActivity.this, "등록해제가 완료되었습니다.", "확인", new DialogInterface.OnClickListener() {  @Override  public void onClick(DialogInterface dialog, int which) {  finish();  }  });  } });… |

※ 해제 단계시 필요한 메소드를 호출 합니다. mIOPLib.reqAuth(); 자세한 내역은 별첨으로 제공된 API Reference를 참고해 주세요.