

Table of Contents

Introduction	1.1
라이브러리 설정	1.2
build.gradle 의존성 설정	1.2.1
AndroidManifest.xml 작성	1.2.2
Client 설정	1.2.3
PlatformContext 구성	1.2.4
EgClient 메뉴얼	1.3
EgClient 생성	1.3.1
Action	1.3.2
ActionListener 구현 및 등록	1.3.3
EgClient 그외 함수	1.3.4
유저연동	1.4
세션 매니저 생성	1.4.1
세션 생성(게스트 로그인) 및 세션 갱신	1.4.2
세션, 유저 정보 조회하기	1.4.3
로그아웃 하기	1.4.4
SNS 계정연동 구현하기	1.4.5

estgames-common-framework (이스트 게임즈 안드로이드 공통 라이브러리)

이스트 게임즈 안드로이드 공통 라이브러리 관련 메뉴얼입니다.

해당 라이브러리는 공통적인 기능과 유저계정 관련한 기능을 담고 있습니다.

공통적인 기능은 배너, 이용약관, 권한, 공지사항, 고객센터, 이벤트, 나라, 언어, 로컬 관련 기능을 처리합니다.

유저 계정은 게스트 로그인, Sns로그인 관련한 기능을 처리합니다.

보시고 잘 모르시겠는 거는 웹플랫폼팀에 문의 주시면 됩니다.

라이브러리 설정

해당 라이브러리를 사용하기 앞서 몇가지 초기설정이 필요합니다. 해당 장에서 순서대로 필요한 설정을 설명합니다.

- [build.gradle 파일 dependency 설정](#)
- [AndroidManifest.xml 작성](#)
- [Client 설정](#)
- [PlatformContext 구성](#)

gradle 파일 의존성 설정

안드로이드 프로젝트는 기본적으로 gradle을 기반으로 프로젝트를 구성합니다. EGMP SDK를 사용하기 위해 build.gradle 파일에 의존성 정보를 등록합니다.

- 해당 라이브러리는 코틀린로 작성되었습니다. 코틀린 플러그인을 적용해주세요.
- Application 클래스 작성에 필요한 라이브러리
 - 'com.android.support:multidex:1.0.1'
- SNS 계정연동에 사용되는 라이브러리
 - 'com.google.android.gms:play-services-auth:15.0.1'
 - 'com.facebook.android:facebook-login:[4, 5)'
- EstGame 라이브러리(제공되는 aar 파일을 app/libs 폴더에 넣어주세요.)
 - implementation 'com.estgame.framework:mp-aos-sdk-release:1.0@aar'
- defaultConfig에 applicationId에 있는 패키지명을 해당게임에 맞는 패키지명을 일치 시켜주어야 구글 로그인이 가능합니다. 자세한 패키지명은 웹플랫폼팀에 문의 부탁드립니다.

build.gradle

```

defaultConfig {
    applicationId "com.estgames.cabalmkor" //게임마다 앱아이디가 다를 수 있습니다. 웹플랫
    품팀의 문의해주세요 .

}

dependencies {
    .
    .
    .
    implementation 'com.android.support:multidex:1.0.1'
    implementation 'com.google.android.gms:play-services-auth:15.0.1'
    implementation 'com.facebook.android:facebook-login:[4, 5)'
    implementation 'com.estgame.framework:mp-aos-sdk-release:1.0@aar'
}
//app/libs 폴더에 넣은 aar파일을 읽기 위한 설정
repositories {
    flatDir {
        dirs 'libs'
    }
}

```


AndroidManifest.xml 설정

안드로이드 앱에서 MP SDK 의 라이브러리를 사용하기 위해서는 안드로이드의 Manifest.xml 파일에 몇가지 설정이 추가 되어야 합니다.

package name 설정

사용자 계정을 외부 OPEN ID 서비스의 계정을 사용하여 연동할 경우 각 프로바이더들의 클라이언트 설정에 package name 을 필요로 합니다. IDE 툴을 사용하여 프로젝트를 생성했을 경우 package name 을 수정할 필요는 없습니다.

⚠ 일부 OPEN ID 서비스의 경우 (google+) package name 정보가 일치하지 않는 경우 계정 연동에 실패합니다.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="net.sample.android.app">
    ...
</manifest>
```

users-permission 설정

EGMP 서비스들은 모두 웹 기반의 API 입니다. 따라서 모바일 앱에서 인터넷 및 네트워크 관련 권한설정을 해야 합니다.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```

Facebook 로그인 설정 (Option)

Android App 에서 Facebook 계정으로 로그인 연동을 하는 경우 Manefest.xml 파일에 페이스북 관련 설정을 해야합니다. 페이스북 로그인은 페이스북 로그인 창을 위한 Activity 등록과 페이스북 앱 인증을 위한 meta-data 설정이 필요합니다. facebook app id 등의 리소스를 분리할 경우 res/values/string.xml 파일에 리소스를 등록해줍니다.

Facebook Login Activity 등록

```
<manifest ... >
    <application ...>
        <activity
            android:name="com.facebook.FacebookActivity" android:exported="true">
            <intent-filter>
```

```
<action android:name="android.intent.action.VIEW" />
<category android:name="android.intent.category.DEFAULT" />
<category android:name="android.intent.category.BROWSABLE" />
<data android:scheme="@string/fb_login_protocol_scheme" />
</intent-filter>
</activity>
</application>
</manifest>
```

Facebook meta-data 설정

```
<manifest ... >
    <application ...>
        <meta-data
            android:name="com.facebook.sdk.ApplicationId"
            android:value="@string/facebook_app_id" />
    </application>
</manifest>
```

Facebook 설정을 위한 리소스 등록 (res/values/string.xml)

fb_login_protocol_scheme 값은 fb + [facebook_app_id] 형태로 입력해주시면 됩니다.

예) fb1234567893456

```
<resource>
    <string name="facebook_app_id">[facebook_app_id]</string>
    <string name="fb_login_protocol_scheme">fb[facebook_app_id]</string>
</resource>
```

Client 설정정보 import

EGMP SDK 는 res/raw 디렉토리에 위치한 설정 파일을 읽어 framework 초기화를 실행합니다. 따라서 SDK에서 필요한 설정 파일을 해당 디렉토리(res/raw) 아래에 위치시켜야 합니다. 필요한 파일은 다음과 같습니다.

- egconfiguration.json

설정 파일은 MP Console 에서 다운로드를 제공할 예정입니다.

```
{  
    "Client" : {  
        "ClientId": "0eae170b-ec59-3f85-bfda-a8c80fc1a3fe.mp.estgames.com",  
        "Secret": "13949b24f1fd9d3c81aab11295c28507ee0a977e9dc5dc93a3bea86f8243b46",  
        "Region": "cm.global.stage"  
    },  
  
    "AccountProviders": {  
        "Google": {  
            "Permissions": "email, profile, openid"  
        },  
        "Facebook": {  
            "Permissions": "public_profile, email"  
        }  
    }  
}
```

PlatformContext 구성

EGMP 서비스를 이용하기 위해 MP Configuration을 초기화 하고 Context를 구성하도록 합니다. Application 클래스를 MP Context로 인식 할 수 있도록 PlatformContext 인터페이스를 구현합니다.

Application 클래스 추가

Application.java

```
public class Application extends MultiDexApplication implements PlatformContext {
    private PlatformContext delegateContext;

    @Override
    public Configuration getConfiguration() {
        return delegateContext.getConfiguration();
    }

    @Override
    public String getDeviceId() {
        return delegateContext.getDeviceId();
    }

    @Override
    public SessionRepository getSessionRepository() {
        return delegateContext.getSessionRepository();
    }

    @Override
    public void onCreate() {
        super.onCreate();
        this.delegateContext = new GenericPlatformContext(getApplicationContext());
    }
}
```

Application 클래스 등록

초기화 코드를 담고 있는 Application 를 앱이 인식 할 수 있도록 AndroidManifest.xml 파일에 등록해줍니다.

AndroidManifest.xml

```
<manifest ... >
    <application
        android:name=' .Application'
        ... >
    ...
</application>
```

```
</manifest>
```

Configuration 속성 관리

`Configuration.Option` 클래스를 이용해 설정값을 등록 할 수 있습니다. 이 옵션을 사용하면 `egconfiguration.json` 파일에 기술된 속성과 다른 값을 적용하여 설정을 구성 할 수 있습니다.

```
Configuration.Option option = new Configuration.Option()
    .clientId("other-client-id")
    .secret("other-client-secret")
    .region("test.region");

delegateContentxt = new AwsPlatformContext(getApplicationContext(), option);
```

`Configuration.Option` 클래스 API

- `Configuration.Option` 의 속성설정 메소드들은 모두 메소드 체인으로 연결됩니다.

메소드 이름	데이터 타입	설명
clientId	String or LazyOption<String>	Client ID 속성을 설정합니다.
secret	String or LazyOption<String>	Secret 속성을 설정합니다.
region	String or LazyOption<String>	Region 속성을 설정합니다.

`LazyOption` 으로 등록된 속성값들은 해당 속성값이 로드 된 후에 속성값에 접근 할 수 있도록 해줍니다. 네트워크등을 통하여 속성값들을 불러와 설정하는 경우에 유용하게 사용할 수 있습니다.

```
Configuration.Option option =
    new Configuration.Option()
        .region(new LazyOption<String>() {
            @Override public String value() {
                // 서버로부터 region값을 로드하는 코드
                String region = ....
                return region
            }
        });
    };
```

Configuration LazyInitializer

`Configuration.Option` 이외에도 `Initializer` 인터페이스를 이용해 Client 설정을 지정할 수 있습니다. 특히 `LazyInitializer` 클래스를 이용해 초기화 시점을 지연 및 컨트롤 할 수 있습니다.

`Initializer` 인터페이스 API

메소드 이름	설명
<code>public Configuration getConfiguration()</code>	

<code>public Configuration getConfiguration()</code>	
<code>public void init(Configuration.Option option)</code>	Configuration.Option 객체를 이용해 Configuration 을 초기화

EgClient 메뉴얼

이용약관, 배너, 공지... 등 게임에서 사용되는 공통 모듈은 EgClient 클래스에 정의되어있습니다.

EgClient 객체에서 각각의 팝업창으로 나오는 기능에 대해서는 Action 객체를 생성처리하며 나라, 국가관련 정보는 문자열로 리턴합니다.

Action 객체에서는 팝업창 관련 클래스입니다. 해당 객체에서는 일어나야할 동작을 등록하고 기능을 실행합니다.

ActionListener 인터페이스에서는 팝업창에서 성공, 취소, 에러시 일어나야 할 동작을 구현합니다.

이 장은 아래와 같은 내용으로 구성됩니다.

- EgClient 생성합니다.
- 각각의 팝업에 대한 Action을 생성하고 실행합니다.
- ActionListener 구현하고 Action에 등록합니다.
- EgClient 그외 함수들의 대해 알아봅니다.

객체 생성

EgClient 에서는 원하는 기능들에 대한 Action 들을 생성합니다. 여기서는 EgClient 를 초기화하는 함수를 알아봅니다.

객체 생성 함수

메소드 이름	리턴 값	매개변수
from	com.estgames.framework.EgClient	android.content.Context

예)

```
EgClient client = EgClient.from(getApplicationContext())
//or
EgClient client = EgClient.from(getApplicationContext())
```

Action

Action 은 해당 라이브러리에서 제공하는 팝업창 동작에 대한 클래스입니다. 앞에서 생성한 EgClient 에서 원하는 팝업창에 대한 Action 을 생성합니다.

EgClient에서 Action 클래스를 만드는 함수

- 해당 함수들은 공통적으로 com.estgames.framework.Action 클래스를 리턴한다.
- android.app.Activity 을 매개변수로 받는다. 해당 팝업을 띄울 액티비티를 매개변수로 받는다.
- Action 클래스에 go 함수로 팝업창을 띄울 수 있다.

메소드 이름	설명
starting	API에서 지정된 순서대로 팝업창을 띄우는 액션
authority	권한
banner	배너
policy	이용약관
notice	공지사항
cs	고객센터
event	이벤트

예)

```
Action<String> starting = client.starting(this);
starting.go();
```

ActionListener 구현 및 등록

ActionListener 에서는 팝업창에서 정상종료, 취소, 에러시 일어나야할 작업을 구현한다.

이름	설명	매개변수
onDone	정상종료	성공 결과값(타입은 제너릭), 팝업마다 결과타입이 다르다.
onCancel	X버튼 혹은 취소	
onError	에러	에러핸들러로 에러종류를 구분합니다.

예)

```
Action<String> starting = client.starting(this);

starting.setListener(new ActionListener<String>() {
    @Override
    public void onDone(String response) {
        // 프로세스 완료 핸들러 작성
    }

    /** 생략가능 ***/
    @Override
    public void onCancel() {
        // 프로세스 취소 핸들러 작성
    }

    /** 생략가능 ***/
    @Override
    public void onError(Throwable t) {
        // 프로세스 에러 핸들러 작성
    }
});

starting.go();
```

EgClient 클래스의 그외 함수

함수명	리턴타입	설명
setLang	void	라이브러리 언어를 설정한다. 한글, 영어 지원
getLang	String	설정된 언어가 없을 경우 기기에 설정된 언어가 리턴된다.
getLocale	java.util.Locale	로케일 정보를 가져온다.
getNation	String	국가 정보 리턴
getSessionManager	SessionManager	세션매니저 생성

유저 연동

유저연동에 가장 먼저할 작업은 `SessionManager` 를 생성하는 일입니다. 매니저에서는 게스트 유저로의 로그인, 유저정보(세션) 조회, 로그아웃(세션종료)을 처리합니다.

`SessionManager` 에서 제공하는 로그인, 로그아웃 메소드는 `Task` 라는 클래스를 리턴하고 이 클래스는 빌더 패턴을 이용하여 해당 작업 종료 혹은 에러 시 이후 처리를 구현합니다.

Sns 연동은 `SignInControl` 에서 처리하며 라이브러리에서 제공하는 버튼을 사용하여 화면을 만들거나 기본으로 제공하는 로그인 팝업창을 사용하여 로그인을 처리합니다.

이 장은 아래내용으로 구성됩니다.

- 세션 매니저 생성합니다.
- 세션 생성(게스트 로그인) 및 세션 갱신
- 세션, 유저 정보 조회하기
- 로그아웃 하기
- SNS 계정연동 구현하기

세션 매니저 생성

EgClient의 `getSessionManager()` 메소드를 사용하여 `SessionManager` 을 생성합니다.

`SessionManager` 에서는 다음과 같은 일을 합니다.

- 게스트 로그인
- 세션, 유저정보 조회
- 로그아웃

```
final SessionManager sessionManager = client.getSessionManager();
```

SessionManager에서 제공하는 메소드

함수명	설명	리턴값	매개변수
isSessionOpen	세션유무 확인	boolean	없음
open	게스트 생성, 계정이 있을 경우 토큰 갱신	com.estgames.framework.core.Task	없음
create	게스트 계정 생성	com.estgames.framework.core.Task	없음
resume	토큰 갱신	com.estgames.framework.core.Task	없음
getToken	토큰정보	com.estgames.framework.session.Token	없음
getProfile	유저관련 정보	com.estgames.framework.session.Profile	없음
signOut	로그아웃	com.estgames.framework.core.Task	없음

세션 생성(게스트 로그인) 및 세션 갱신

세션 생성은 `create()` 메소드로 만드며 해당 메소드는 Task 클래스를 리턴합니다. Task는 빌더 패턴을 이용하여 로그인이 끝난 이후 혹은 에러 시 처리를 구현합니다.

앱을 시작했을 때 세션이 없을 경우에는 `create()` 메소드를 사용하여 게스트 계정을 만들 수 있고 현재 세션이 있는 상태라면(SNS계정 포함) 세션을 다시 사용하기 위해 갱신을 해주어야 합니다. 갱신은 `resume()` 메소드를 호출하여 토큰을 갱신할 수 있습니다.

Task에서 제공하는 메소드에서는 하나의 함수를 구현하는 인터페이스인 `Acceptor`를 매개변수로 받습니다.

Task에서 제공하는 메소드

함수명	설명	리턴값	매개변수
<code>asyncAccept</code>	정상종료 이후 처리	<code>com.estgames.framework.core.Task</code>	<code>Acceptor<T></code>
<code>onError</code>	에러시 처리 구현	<code>com.estgames.framework.session.Token</code>	<code>Acceptor<T></code>

예) 게스트 세션생성 및 생성 이후, 실패 시 처리

```
sessionManager
    .create()
    .onError(new Task.Acceptor<Throwable>() {
        @Override
        public void accept(Throwable t) {
            // 세션 생성 실패 핸들러 작성
        }
    })
    .asyncAccept(new Task.Acceptor<String>() {
        @Override
        public void accept(String token) {
            // 세션 생성 완료 핸들러 작성
        }
    });
}
```

예) 세션 갱신

```
sessionManager
    .resume()
    .onError(new Task.Acceptor<Throwable>() {
        @Override
        public void accept(Throwable throwable) {
            currentTitle.setText("세션 resume 실패 = " + throwable.getMessage());
        }
    });
}
```

```
.asyncAccept(new Task.Acceptor<String>() {
    @Override
    public void accept(String s) {
        currentTitle.setText("세션 resume 성공 = " + s);
    }
});
```

세션, 유저정보 조회하기

SessionManager에서 getToken() 메소드로 현재 세션의 토큰 정보를 조회할 수 있으며 getProfile() 메소드로 프로필 정보를 조회할 수 있습니다.

SessionManager 세션정보 조회관련 메소드

함수명	리턴타입	설명
getToken	com.estgames.framework.session.Token	토큰정보
getProfile	com.estgames.framework.session.Profile	유저관련 정보

Token

함수명	리턴타입	설명
getEgToken	String	EgToken
getRefreshToken	String	RefreshToken

Profile

함수명	리턴값	설명
getEgId	String	EgId
getUserId	String	유저아이디
getProvider	String?	프로바이더(gmail,facebook)
getEmail	String?	이메일
getPrincipal	String	Principal

예)

```

Token token = client.getSessionManager().getToken(); // 현재 세션의 토큰정보 조회
Profile profile = client.getSessionManager().getProfile(); // 현재 세션의 프로필정보 조회
하기

if (token != null) {
    egTokenTitle.setText("getEgToken = " + token.getEgToken());
    refreshTokenTitle.setText("getRefreshToken = " + token.getRefreshToken());
}

if (profile != null) {
    egIdTitle.setText("getEgId = " + profile.getEgId());
    emailTitle.setText("getEmail = " + profile.getEmail());
    principalTitle.setText("getPrincipal = " + profile.getPrincipal());
}

```

```
    providerTitle.setText("getProvider = " + profile.getProvider());
    userIdTitle.setText("getUserId = " + profile.getUserId());
}
```

로그아웃하기

SessionManager 에 있는 **signOut** 메소드로 로그아웃을 진행합니다.

```
sessionManager.signOut().asyncAccept(new Task.Acceptor() {  
    @Override public void accept(Object o) {  
        // 세션 종료 핸들러 작성  
        System.out.println("로그아웃");  
    }  
});
```

SNS 계정연동

플랫폼 SDK는 SNS 계정에 로그인 하고 플랫폼 계정과 연동하는 역할을 하는 `SignInControl` 클래스를 제공합니다.

현재 플랫폼이 제공하는 SNS 계정연동은 Facebook 과 Google 입니다. SNS 계정연동은 Android Framework에서 제공하는 순수 컴포넌트만을 이용할 수도 있고, EG 플랫폼에서 제공하는 로그인 버튼 컴포넌트나 로그인 대화창 (Activity 기반)을 이용할 수도 있습니다.

계정연동은 다음과 같은 순서로 할 수 있습니다.

기본 버튼으로 로그인

1. 레이아웃에 버튼 배치

```
<Button
    android:id="@+id	btn_facebook_login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="5dp"
    android:text="Facebook Login"/>
```

2. SignInControl 설정 (in Activity)

`SignInControl` 객체는 `onActivityResult` 콜백에 등록해야 합니다. 따라서 인스턴스 변수로 생성해야 합니다.

```
// Sign in option 설정
SignInControl.Option option = new SignInControl.Option()
    .setSignInResultHandler(new SignInResultHandler() {
        @Override
        public void onComplete(Result.Login result) {
            // 로그인 성공 핸들러 작성
        }

        @Override
        public void onError(Throwable t) {
            // 로그인 실패 핸들러 작성
        }

        @Override
        public void onCancel() {
            // 로그인 취소 핸들러 작성
        }
    });
});
```

```
// SignInControl 객체생성
SignInControl signInControl = SignInControl.createControl(activity, option);
```

3. Button 핸들러에 연결

로그인 버튼을 눌렀을때 로그인을 시도하도록 핸들러에 등록합니다.

```
Button facebookLogin = findViewById(R.id.btn_facebook_login);
facebookLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        signInControl.signIn(Provider.FACEBOOK, activity);
    }
});
```

4. ActivityResult 핸들러 작성

로그인 결과를 핸들러에 전달 할 수 있도록 Activity.onActivityResult 메소드를 작성합니다.

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    signInControl.onActivityResult(requestCode, resultCode, data);
}
```

EG SDK 제공 버튼으로 로그인

1. 레이아웃에 버튼 배치

```
<com.estgames.framework.ui.buttons.FacebookSignInButton
    android:id="@+id/btn_facebook_login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>

<com.estgames.framework.ui.buttons.GoogleSignInButton
    android:id="@+id/btn_google_login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```

2. SignInControl 설정 (in Activity)

`SignInControl` 객체는 `onActivityResult` 콜백에 등록해야 합니다. 따라서 인스턴스 변수로 생성해야 합니다.

레이아웃 xml 파일에 명시한 `SignInButton` 을 Control 객체와 연결합니다.

```
// Sign in option 설정
```

```

SignInControl.Option option = new SignInControl.Option()
    .addSignInButton((SignInButton) findViewById(R.layout.btn_facebook_login))
    .addSignInButton((SignInButton) findViewById(R.layout.btn_google_login))
    .setSignInResultHandler(new SignInResultHandler() {
        @Override
        public void onComplete(Result.Login result) {
            // 로그인 성공 핸들러 작성
        }

        @Override
        public void onError(Throwable t) {
            // 로그인 실패 핸들러 작성
        }

        @Override
        public void onCancel() {
            // 로그인 취소 핸들러 작성
        }
    });

// SignInControl 객체생성
SignInControl signInControl = SignInControl.createControl(activity, option);

```

3. ActivityResult 핸들러 작성

로그인 결과를 핸들러에 전달 할 수 있도록 Activity.onActivityResult 메소드를 작성합니다.

```

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    signInControl.onActivityResult(requestCode, resultCode, data);
}

```

EG SDK 제공 대화창(SignInActivity)으로 로그인

1. SignInControl 설정 (in Activity)

`SignInControl` 객체는 `onActivityResult` 콜백에 등록해야 합니다. 따라서 인스턴스 변수로 생성해야 합니다.

레이아웃 xml 파일에 명시한 `SignInButton` 을 Control 객체와 연결합니다.

```

// Sign in option 설정
SignInControl.Option option = new SignInControl.Option()
    .setSignInResultHandler(new SignInResultHandler() {
        @Override
        public void onComplete(Result.Login result) {
            // 로그인 성공 핸들러 작성
        }
    });

```

```
@Override  
public void onError(Throwable t) {  
    // 로그인 실패 핸들러 작성  
}  
  
@Override  
public void onCancel() {  
    // 로그인 취소 핸들러 작성  
}  
});  
  
// SignInControl 객체생성  
SignInControl signInControl = SignInControl.createControl(activity, option);
```

2. ActivityResult 핸들러 작성

로그인 결과를 핸들러에 전달 할 수 있도록 Activity.onActivityResult 메소드를 작성합니다.

```
@Override  
public void onActivityResult(int requestCode, int resultCode, Intent data) {  
    signInControl.onActivityResult(requestCode, resultCode, data);  
}
```

3. 로그인 대화창 호출

```
signInControl.signIn(activity);
```