Ad@m Android Publisher SDK Guide

이 가이드는 Android Application 에 모바일 광고를 노출하기 위한 광고 데이터요청과 처리 방법을 설명합니다.

사이트/앱 운영정책에 어긋나는 경우 적립금 지급이 거절 될 수 있으니 유의하시기 바랍니다.

- 문의 고객센터 http://cs.daum.net/mail/form/256.html
- 사이트/앱 운영 정책
  http://mobile.biz.daum.net/guide/guide_siteapp_policy.jsp

_

최근 변경이력

v2.3.1

Bugfix

- 일부 앱에서 TransactionTooLargeException 가 발생하는 오류를 수정

v2.3.0

New

- IDE에서 Layout 작업시에 광고 영역을 노출함(isInEditMode 지원)
- 처리할 수 없는 URL 중에 intent scheme인 경우에는 마켓으로 이동함.
- Google Advertising ID 추출 로직 추가.

Change

- 기본 배너 크기를 320x48에서 320x50으로 변경.
- 프로젝트 빌드 스크립트 수정(Google Play Service SDK를 Dependency에 추가)

Bugfix

- 가이드 앱의 빌드 SDK 버전을 8에서 7로 낮춤.
- Sample 앱 레이아웃 일부 수정.
- 테스트 앱의 빌드 SDK 범위를 8에서 7로 변경.

Ad@m 광고 삽입 방법

Ad@m SDK 구성

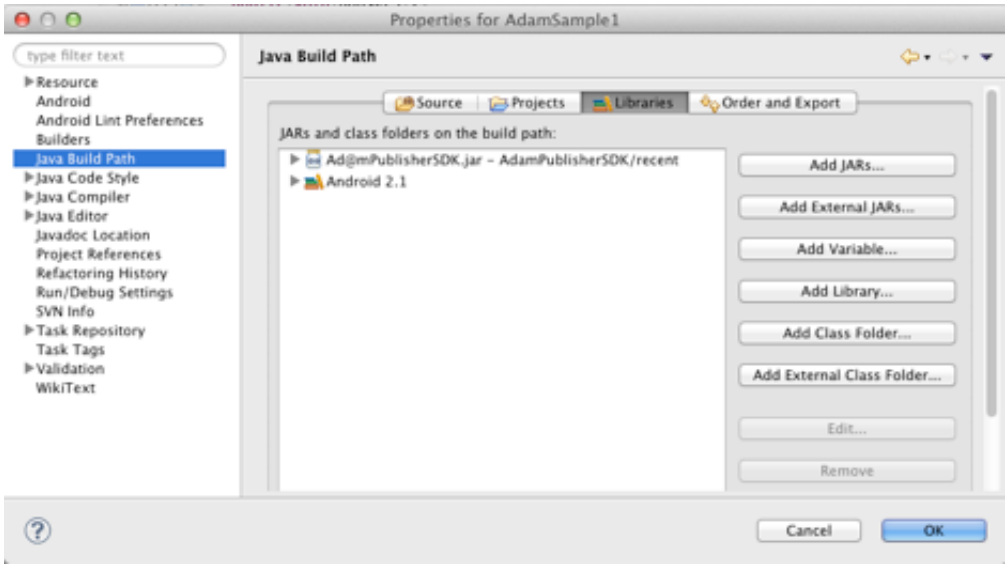
- Ad@mPublisherSDK.jar : Ad@m 광고를 삽입해주는 라이브러리
- Sample/AdamSample/src/net/daum/adam/publisher/sample/
    - BannerTypeXML1.java : 광고를 xml 로 붙인 샘플
    - BannerTypeXML2.java : 광고 Visible 처리 및 pause, resume 처리 예시 샘플
    - BannerTypeJava.java : 광고를 java 코드로 붙인 샘플
    - InterstitialActivity.java : Interstitial(전면형) 광고를 java 코드로 붙인 샘플

1 단계 : client ID 발급받기

실제 광고를 다운로드 받고, 수익창출을 위해서 mobile.biz.daum.net 에서 사이트/앱 등록 후 client ID 를 발급받아야 한다. 아래 URL 을 통해 애플리케이션을 등록할 수 있다. http://mobile.biz.daum.net/guide/guide_siteapp1.jsp

2 단계 : 라이브러리 import

Ad@mPublisherSDK 를 프로젝트 내에 라이브러리로 Import 한다. (Ad@m Publisher SDK 2.0 부터는 **Android 2.1(API Level 7)** 이상의 환경에서 동작한다.)



Google Play Store에 App을 개시하는 경우, App 내에 광고가 있다면 반드시 Google Advertising ID를 사용하도록 규정이 변경되었다.

이에 따라, SDK 2.3.0 부터는 App에서 Google Play Service SDK를 사용할 수 있는 경우에 한해 Google Advertising ID를 사용할 수 있도록 기능이 추가되었다.

만약 앱에 Ad@m 광고를 넣어서 Google Play Store에 개시하고 있다면 **반드시 SDK 2.3.0 이후 버전을 사용**해야 한다.

Google Play Service SDK를 사용하기 위해서는 아래 경로에 있는 JAR 파일을 프

로젝트 내 libs/ 경로에 복사해야 한다.

```
<android-sdk>/extras/google/google_play_services/libproject/google
```

App에서 Proguard를 사용하고 있다면, 반드시 아래 내용을 추가로 넣어주어야 한다.

```
-keep class * extends java.util.ListResourceBundle {
    protected Object[][] getContents();
}

-keep public class
com.google.android.gms.common.internal.safeparcel.SafeParcelable
 {
    public static final *** NULL;
}

-keepnames @com.google.android.gms.common.annotation.KeepName
class *
-keepclassmembernames class * {
    @com.google.android.gms.common.annotation.KeepName *;
}

-keepnames class * implements android.os.Parcelable {
    public static final ** CREATOR;
}
```

Google Play Service SDK와 관련해 보다 자세한 사항은 [Setting Up Google Play Services][1] 링크를 참고하기 바란다.

3 단계 : AndroidManifest.xml 설정

- 아래 세 가지 필수 권한을 AndroidManifist.xml 에 추가한다.

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

SDK 2.3.0 부터는 최소한의 권한으로 INTERNET, ACCESS_NETWORK_STATE 권한을 설정해야 한다. 필수 권한 미 설정시 정상적 광고 노출 되지 않는다.

- 광고를 넣을 Activity 에 반드시 android:configChanges="orientation" 을 설정해준다.
- Interstitial(전면형) 광고를 추가하기 위해서는 반드시 아래 명시된 Activity 를 추가해야 한다.

AndroidManifest.xml

```
<application
    android:icon="@drawable/icon"
    android:label="@string/appName" >

    <!-- Google Play Service SDK 설정 -->
    <!-- Google Play Service SDK를 사용하는 App에 한해 아래 meta-
data 테그를 추가한다. -->
    <!-- (https://developer.android.com/google/play-
services/setup.html) -->
    <meta-data
android:name="com.google.android.gms.version"

android:value="@integer/google_play_services_version" />

    <activity
```

```
        android:name=".TestAppActivity"
        android:configChanges="orientation|keyboardHidden"
        android:label="@string/appName" >
        <intent-filter>
            <action
android:name="android.intent.action.MAIN" />
            <category
android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <!-- Interstitial 광고를 사용하기 위해서는 반드시 이 부분을 추가해야
한다. -->
    <activity

android:name="net.daum.adam.publisher.impl.AdInterstitialActi

        android:configChanges="orientation|keyboardHidden"
        android:screenOrientation="portrait" />

    <!-- 광고를 노출할 Activity 에
android:configChanges="orientation"을 반드시 추가해야 한다. -->
    <activity
        android:name=".BannerActivity"
        android:configChanges="orientation|keyboardHidden"
/>
</application>

<!-- 아래 권한을 반드시 추가해야 한다. -->
<uses-permission android:name="android.permission.INTERNET"
/>
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
```

4 단계 : 광고 요청을 위한 UI 구성 및 설정

4-a. Xml 방식

- Layout 의 main.xml 에서 광고가 노출되고자 하는 곳에 AdView 객체를 추가한다.
- *광고를 노출 가능한 최소크기(320x50)보다 작게 광고 뷰가 할당되는 경우에는 광고가 노출되지 않을 수 있다.*
- 그 이외의 속성 값은 어플리케이션의 특성에 따라 자유롭게 변경 가능하다.

res/layout/main.xml

```
<RelativeLayout

xmlns:app="http://schemas.android.com/apk/res/[APP_PACKAGENAME]"

    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <!-- 광고를 사용하기 위해서는 반드시 Client ID 를 발급받아 사용해야 한다. -->
    <net.daum.adam.publisher.AdView
        android:id="@+id/adview"
        android:visibility="invisible"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        clientId=”TestClientId”
        requestInterval=”60”/>
</RelativeLayout>
```

위 레이아웃에 설정한 AdView 객체를 Activity 에서 사용하는 방법을 아래 예를 통해 살펴보도록 하자.

SDK 2.0 부터는 AdHttpListener 를 반드시 구현할 필요가 없고, 필요한 경우에 해당 Listener 를 구현해서 설정해주면 된다.

현재 5 개의 Listener 를 지원하고 있으며, 자세한 내역은 아래 예제 코드와 Class Reference 를 통해 살펴보도록 하자.

[YourApplication]Activity.java

```
public class BannerTypeXML1 extends Activity {
    private static final String LOGTAG = "BannerTypeXML1";
    private AdView adView = null;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.adam_sample_1);
        initAdam();
    }



    @Override
    public void onDestroy() {
        super.onDestroy();

        if (adView != null) {
            adView.destroy();
            adView = null;
        }
    }

    private void initAdam() {
        // Ad@m sdk 초기화 시작
```

```
        adView = (AdView) findViewById(R.id.adview);

        // 광고 리스너 설정

        // 1. 광고 클릭시 실행할 리스너
        adView.setOnAdClickedListener(new
OnAdClickedListener() {
            @Override
            public void OnAdClicked() {
                Log.i(LOGTAG, "광고를 클릭했습니다.");
            }
        });

        // 2. 광고 내려받기 실패했을 경우에 실행할 리스너
        adView.setOnAdFailedListener(new OnAdFailedListener()
{
            @Override
            public void OnAdFailed(AdError error, String
message) {
                Log.w(LOGTAG, message);
            }
        });

        // 3. 광고를 정상적으로 내려받았을 경우에 실행할 리스너
        adView.setOnAdLoadedListener(new OnAdLoadedListener()
{
            @Override
            public void OnAdLoaded() {
                Log.i(LOGTAG, "광고가 정상적으로 로딩되었습니다.");
            }
        });

        // 4. 광고를 불러올때 실행할 리스너
```

```
        adView.setOnAdWillLoadListener(new
OnAdWillLoadListener() {
            @Override
            public void OnAdWillLoad(String url) {
                Log.i(LOGTAG, "광고를 불러옵니다. : " + url);
            }
        });


        // 5. 전면형 광고를 닫았을때 실행할 리스너
        adView.setOnAdClosedListener(new OnAdClosedListener()
{
            @Override
            public void OnAdClosed() {
                Log.i(LOGTAG, "광고를 닫았습니다.");
            }
        });


        // 할당 받은 clientId 설정
        // adView.setClientId(“TestClientId”);


        // 광고 갱신 주기를 12초로 설정
        // adView.setRequestInterval(12);


        // 광고 영역에 캐시 사용 여부 : 기본 값은 true
        adView.setAdCache(false);

        // Animation 효과 : 기본 값은 AnimationType.NONE

adView.setAnimationType(AnimationType.FLIP_HORIZONTAL);
```

```
            adView.setVisibility(View.VISIBLE);
        }
    }
```

광고 영역은 웹뷰를 사용하고 있고, 기본적으로 캐시를 사용하고 있다. 만약 캐시를 사용하지 않을 경우에는 위 예제와 같이 `adView.setAdCache(false);` 를 호출해 캐시를 사용하지 않도록 설정할 수 있다. 이 경우에는 기존에 캐시 영역의 데이터를 모두 삭제한다.

AdView 클래스에는 위와 같이 5 개의 리스너를 제공하고 있다.

- AdView.OnAdClickedListener : 광고 클릭할 경우 실행할 리스너
- AdView.OnAdFailedListener : 광고 내려받기 실패할 경우 실행할 리스너
- AdView.OnAdLoadedListener : 광고가 내려받았을 경우 실행할 리스너
- AdView.OnAdWillLoadListener : 광고를 불러오기 전에 실행할 리스너
- AdView.OnAdClosedListener : 전면형 광고를 닫을 때 실행할 리스너

위 예제에서는 현재 5 개의 리스너를 설정하고 있지만, 리스너가 필요가 없으면 굳이 설정하지 않아도 된다. 리스너와 관련된 자세한 내역은 클래스 레퍼런스를 통해 살펴보도록 하자.

4-b. Java 방식

광고를 넣고자 하는 view 가 들어 있는 Activity 가 생성될 때 AdView 객체를 생성하고 광고 요청을 위해 광고 View 에 필요한 리스너와 할당 받은 ClientId 를 설정한다. XML 레이아웃을 이용해 광고 생성할 때와 거의 동일하다.

```
public class BannerTypeJava extends Activity {
    private static final String LOGTAG = "BannerTypeJava";
    private RelativeLayout relativeLayout = null;
```

```
    private AdView adView = null;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        relativeLayout = new RelativeLayout(this);

        // Ad@m 광고 뷰 생성 및 설정
        adView = new AdView(this);

        // 광고 클릭시 실행할 리스너
        adView.setOnAdClickedListener(new
OnAdClickedListener() {
            @Override
            public void OnAdClicked() {
                Log.i(LOGTAG, "광고를 클릭했습니다.");
            }
        });

        // 광고 내려받기 실패했을 경우에 실행할 리스너
        adView.setOnAdFailedListener(new OnAdFailedListener()
{
            @Override
            public void OnAdFailed(AdError arg0, String arg1)
{
                Log.w(LOGTAG, arg1);
            }
        });

        // 광고를 정상적으로 내려받았을 경우에 실행할 리스너
        adView.setOnAdLoadedListener(new OnAdLoadedListener()
{
```

```
            @Override
            public void OnAdLoaded() {
                Log.i(LOGTAG, "광고가 정상적으로 로딩되었습니다.");
            }
        });

        // 광고를 불러올때 실행할 리스너
        adView.setOnAdWillLoadListener(new
OnAdWillLoadListener() {

            @Override
            public void OnAdWillLoad(String arg1) {
                Log.i(LOGTAG, "광고를 불러옵니다. : " + arg1);
            }
        });

        // 광고를 닫았을때 실행할 리스너
        adView.setOnAdClosedListener(new OnAdClosedListener()
{

            @Override
            public void OnAdClosed() {
                Log.i(LOGTAG, "광고를 닫았습니다.");
            }
        });

        // 할당 받은 clientId 설정
        adView.setClientId("TestClientId");

        // 광고 갱신 시간 : 기본 60초
        adView.setRequestInterval(12);
```

```
        // Animation 효과 : 기본 값은 AnimationType.NONE

adView.setAnimationType(AnimationType.FLIP_HORIZONTAL);

        adView.setVisibility(View.VISIBLE);

        // XML상에 android:layout_alignParentBottom="true" 와 같
은 역할을 함
        RelativeLayout.LayoutParams params = new
RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.MATCH_PA
 RelativeLayout.LayoutParams.WRAP_CONTENT);
        params.addRule(RelativeLayout.ALIGN_PARENT_BOTTOM);

        // 위에서 만든 레이아웃을 광고 뷰에 적용함.
        adView.setLayoutParams(params);

        setContentView(relativeLayout);
    }

    @Override
    public void onDestroy() {
        super.onDestroy();

        if (adView != null) {
            adView.destroy();
            adView = null;
        }
    }
}
```

선택 : Interstitial (전면형) 광고 요청을 위한 설정

Interstitial(전면형) 광고는 당분간 Ad@m 의 네트워크 파트너를 대상으로 노출된다. Ad@m 의 네트워크 파트너가 아닐 경우에도 Expandable(확장형),

Animated Banner (애니메이션형)형의 Rich Media 광고가 노출된다.

Interstitial(전면형) 광고를 넣고자 하는 Activity 가 생성될 때 AdInterstitial 객체를 생성하고 광고 요청을 위해 필요한 리스너와 할당 받은 ClientId 를 설정한다. 이때, 반드시 3 단계에 명시한 XML 코드를 AndroidManifest.xml 에 반드시 추가해야 한다.

```
public class InterstitialActivity extends Activity {
    /** 전면형 광고 선언 */
    AdInterstitial mAdInterstitial = null;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // 1. 전면형 광고 객체 생성
        mAdInterstitial = new AdInterstitial(this);
        // 2. 전면형 광고 클라이언트 ID를 설정한다.

mAdInterstitial.setClientId(“InterstitialTestClientId”);
        // 3. (선택)전면형 광고 다운로드시에 실행할 리스너
        mAdInterstitial.setOnAdLoadedListener(new
OnAdLoadedListener() {
            @Override
            public void OnAdLoaded() {
                Log.i("InterstitialTab", "광고가 로딩되었습니다.");
            }
        });
        // 4. (선택)전면형 광고 다운로드 실패시에 실행할 리스너
        mAdInterstitial.setOnAdFailedListener(new
OnAdFailedListener() {
            @Override
            public void OnAdFailed(AdError error, String
errorMessage) {
```

```
                Toast.makeText(InterstitialActivity.this,
 errorMessage, Toast.LENGTH_LONG).show();
            }
        });
        // 5. (선택)전면형 광고를 닫을 시에 실행할 리스너
        mAdInterstitial.setOnAdClosedListener (new
OnAdClosedListener() {
            @Override
            public void OnAdClosed() {
                Log.i("InterstitialTab", "광고를 닫았습니다. ");
            }
        });
        // 6. 전면형 광고를 불러온다.
        mAdInterstitial.loadAd();
    }
    @Override
    public void onDestroy() {
        super.onDestroy();
        if (mAdInterstitial != null) {
            mAdInterstitial = null;
        }
    }
}
```

추가정보(FAQ)

Q1. 광고 수신이 되지 않을때는 어떻게 하나요?

Ad@m 은 유효 광고의 100% 노출을 보장하지 않습니다. 유효 광고 노출율은 송출 가능한 광고의 총 수량과 광고 호출수에 따라 달라지게 됩니다. 광고의 총 수량은 한정되어 있으나, 이에 비해 광고의 호출수가 많기 때문에 유효 광고의 수신에 실패하는

경우가 자주 발생할 수 있습니다. 또한 시간대나 앱의 종류, 날짜에 따라서도 노출 가능한 광고의 수가 달라질 수 있습니다.

배너 광고의 경우 http://mobile.biz.daum.net 에서 '사이트/앱 등록' 메뉴에 접속하면 '하우스애드'를 등록할 수 있습니다. 하우스애드란 자신의 애플리케이션에 자체 광고를 노출할 수 있는 기능으로, 광고 서버에서 유효 광고를 보내줄 수 없는 경우 자신이 등록한 하우스애드가 수신됩니다.

Interstitial 광고의 경우에는 하우스 애드가 지원되지 않으므로, 일정 시간 이후 다시 호출해야 합니다.

Q2. 인터넷(3G 또는 WIFI)이 연결되지 않을 경우에 어떻게 하나요?

내부적으로 인터넷 연결이 끊기면 광고 송출이 자동으로 중지되고, 연결되면 자동으로 광고 송출을 시작하게 됩니다.

Q3. 광고 영역이 텅 비어보입니다. 아담 버그 아닌가요?

최초 광고를 내려받기 전 까지는 광고 요청에 시간이 걸리기 때문에 잠시 비어있을 수 있습니다. SDK 2.0 부터는 기본적으로 광고를 감싸고 있는 영역이 View.GONE 상태였다가, 광고가 완전히 내려받은 후에 View.VISIBLE 로 바꾸고 있습니다. 한번 View.VISIBLE 로 바뀐 영역은 광고 내려받기가 실패할 지라도 다시 가리지 않습니다.

어떠한 사유로 광고 내려받기가 실패할 경우에는 AdView 객체의 OnAdFailedListener 리스너를 설정해 필요한 기능을 앱에 맞게 설정하시면 됩니다.

Q4. 시스템 앱에서 Expandable 광고가 보이지 않습니다. 왜 그런건가요?

키보드 앱과 같은 시스템 앱에서는 기존 광고 영역을 자유롭게 변경하기 어려운 관계로 Expandable(확장형) 광고를 보여주기 어렵습니다. 따라서 시스템 앱에서는 단순 클릭형 배너만 노출됩니다.

Q5. Build PATH 에 라이브러리가 있는데도 앱 사용시 오류가 납니다.

Android Tools 버전 17 부터는 libs 폴더에 있는 라이브러리는 앱에 자동으로 포함됩니다.[2]

만약 libs 폴더에 있지 않은 경우에는 해당 라이브러리가 Export 되고 있는지 반드시 확인해야 합니다.

Q6. 2.3.0 sdk로 변경만 하면 구글 광고 ID를 sdk 내에서 생성해주는건지 답변부탁드립니다.

SDK 내에서 Google 광고 ID를 생성해주지는 않습니다.

앱에서 Google 광고 ID를 추출할 수 있으면 해당 ID를 사용하는 것 뿐입니다.

앱을 빌드시, 반드시 Google Play Service SDK를 함께 넣어주셔야만 Google 광고 ID를 사용할 수 있습니다.

Google Play Service SDK 설정과 관련해 보다 자세한 사항은 Setting Up Google Play Services 페이지를 참고하면 될 것 같습니다.

1. https://developer.android.com/google/play-services/setup.html?hl=ko ↩

2. http://tools.android.com/recent/dealingwithdependenciesinandroidprojects ↩