Integration und Konfiguration der Android Tracking Library

Version 4.0

Webtrekk
Drive Your Profit.

Inhalt

1 Vorwort

Mit Hilfe der Android Tracking Library ist es möglich Nutzungshäufigkeit und Aktivitäten Ihrer Android Apps in Webtrekk zu erfassen. Dieses Dokument beschreibt die technische Integration des Trackings in Ihre Android App.

Um die Analysemöglichkeiten von Webtrekk optimal einzusetzen, sollten Sie sich in der Konzeptionsphase an unseren Schulungsunterlagen „Grundlagen zur Datenerfassung“ orientieren. Sollte Ihnen das Dokument nicht vorliegen, können Sie sich jeder Zeit an support@webtrekk.com oder Ihre(n) persönlichen Consultant wenden.

Durch den Einsatz der Tracking API werden App-Aufrufe und Aktivitäten des Nutzers als Requests an das Webtrekk Tracking System geschickt.

In Abhängigkeit zu dem Onlinestatus des Nutzers werden die Requests an Webtrekk entsprechend gesendet:

1. Ist das mobile Gerät online, werden die Requests in einem definierbaren Zeitintervall gesendet.
2. Ist das Gerät offline, werden die Aktivitäten zwischengespeichert und versendet, sobald das Gerät wieder online ist.

1.1 Systemvoraussetzungen

Trackbare Systeme: Alle Geräte mit dem Betriebssystem Android Version 4.0 oder höher.

Entwicklungsumgebung: Die Einbindung in die App ist grundsätzlich mit jeder Entwicklungsumgebung möglich, die vorliegende Dokumentation bezieht sich auf Android Studio

1.2 Integration der Tracking Library

Bevor Sie mit dem Tracking Ihrer App starten können, muss die Library in ihr Android Projekt eingebunden werden.

Nachfolgend werden die Schritte aufgezeigt wie dies unter Android Studio mit Gradle als Buildsystem erfolgen kann. Die Pfade beziehen sich dabei auf das Android Default Projekt Layout.

1. Erstellen Sie, falls noch nicht vorhanden, im Hauptordner Ihres zu trackenden Android-Modules einen Ordner „libs“.
2. Kopieren Sie die Datei „WebtrekkSDK-release.aar“ in das Verzeichnis aus Schritt 1
3. Fügen Sie diesen Ordner in der Gradle Build Datei als lokales Repository des Android-Modules hinzu, indem sie in der „build.gradle“ Datei des Moduls folgende Zeilen ergänzen:

```
repositories{
    flatDir{
        dirs 'libs'
    }
}
```

4. Ergänzen sie die .aar Datei als Abhängigkeit, ebenfalls in der build.gradle Datei des Moduls.

```
dependencies {
    ...
    compile(name:'WebtrekkSDK-release', ext:'aar')
    ...
}
```

5. Permissions anpassen:

Für das Tracking werden folgende zwei Permissions immer benötigt, sonst können keine Requests übertragen werden.

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

Wenn Sie den Benutzernamen oder anderen Profilinformationen aus dem Playstore wie die E-Mail Adresse des Nutzers tracken wollen, müssen zusätzlich die folgenden Permissions ergänzt werden:

```
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
```

Element	Erforderlich für
READ_PROFILE	Vorname, Nachname
READ_CONTACTS	E-Mail Adresse
GET_ACCOUNTS	

6. Konfigurationsdatei hinterlegen

Kopieren Sie die Konfigurationsdatei „webtrekk_config.xml“ in den Ressourcen Ordner Ihres Android Modules unter „**module/res/raw/**“. Diese Datei beinhält zum einen die notwendige Konfiguration zum Versenden von Tracking Requests an Ihren Webtrekk Account (siehe

Tracking-Konfiguration) und kann ebenfalls zum Definieren von Parametern und Werten genutzt werden.

2 Tracking-Funktionalität

Die Tracking-Library erlaubt es, Seiten- und Aktionsrequests an Webtrekk zu übermitteln. Genau wie in einer Standard-Webseitenverpixelung können diese Requests durch weitere Parameter angereichert werden.

Das Tracking eines App-Besuchs ist dabei immer analog aufgebaut.

- Initialisieren der Webtrekk-Instanz
- Senden beliebig vieler Tracking-Requests während der Nutzung

Zum Tracken von Activities stehen Ihnen zwei Möglichkeiten zur Verfügung:

Automatisches Tracking: Hierbei werden für die Android Lifecycle-Methoden entsprechende Callbacks in der Android App registriert. Diese werden dann automatisch aufgerufen und tracken alle Aktivitätsaufrufe automatisch.

Manuelles Tracking: Hierbei müssen Sie in jeder zu trackenden Activity das Tracking selbst implementieren.

Auch wenn Sie das automatische Tracking aktivieren, sind Sie nicht gezwungen alle Activities Ihrer Applikation zu messen. Über die Möglichkeiten der Konfigurationsdatei können Sie auch einzelne Activities von automatischen Tracking ausschließen sowie umgekehrt.

Für beide Möglichkeiten benötigen Sie eine Konfigurationsdatei in dem Format XML siehe

Tracking-Konfiguration

Webtrekk empfiehlt das aktivieren des automatischen Trackings und eine vollständige Konfiguration über die Konfigurationsdatei.

2.1 Tracking-Konfiguration

Über die Konfigurationsdatei „webtrekk_config.xml“ hinterlegen Sie alle notwendigen Konfigurationseinstellungen für das messen von Aktivitäten in Ihrer App.

Die folgende Tabelle enthält eine Übersicht aller Konfigurationsmerkmale, welche für das Versenden von Informationen benötigt werden oder welche aktiviert werden können um zusätzliche Daten zum Nutzer zu erheben:

Element	Beschreibung	Optional
version	Versionsnummer der Konfigurationsdate	☒
trackDomain	Ihre Webtrekk Track Domain	☒
trackId	Ihre Webtrekk Track-ID	☒
sampling	Anhand des Sampling Faktors wird nur jeder n-te User erfasst.	☑
sendDelay	Definiert das Intervall indem registrierte Requests an Webtrekk übermittelt werden sollen.	☑
maxRequests	Gibt die maximale Anzahl der Requests an die auf dem Gerät zwischengespeichert werden.	☑
autoTracked	Wenn das automatische Tracking aktiviert ist wird beim Starten einer Activity automatisch ein Request erzeugt	☑
autoTrackAppUpdate	Kapitel 5.1	☑
autoTrackAdvertiserId		☑
autoTrackAppVersionName		☑
autoTrackAppVersionCode		☑
autoTrackAppPreInstalled		☑
autoTrackPlaystoreMail		☑
autoTrackApiLevel		☑
autoTrackScreenOrientation		☑
autoTrackConnectionType		☑
autoTrackAdvertisementOptOut		☑
autoTrackRequestUrlStoreSize		☑
enableRemoteConfiguration	Kapitel 9	☑
trackingConfigurationUrl		☑
resendOnStartEventTime	Definiert den Zeitraum ab wann eine Interaktion zu einer neuen Session gezählt werden soll, wenn der Nutzer zuvor bspw. angerufen wurde	☑

Das folgende Beispiel einer Konfigurationsdatei enthält alle Elemente welche beim Initialisieren der Webtrekk Instanz ausgewertet werden und die gesetzten Default-Werte, falls die Elemente in der Konfigurationsdatei nicht überschrieben wurden:

```
<?xml version="1.0" encoding="utf-8"?>
<webtrekkConfiguration>
    <version>0</version>
    <trackDomain type="text"></trackDomain>
    <trackId type="text"></trackId>
    <sampling type="text">0</sampling>
    <sendDelay type="text">60</sendDelay>
    <maxRequests type="number">5000</maxRequests>

    <autoTracked>true</autoTracked>

    <autoTrackAppUpdate>true</autoTrackAppUpdate>
    <autoTrackAdvertiserId>true</autoTrackAdvertiserId>
    <autoTrackAppVersionName>true</autoTrackAppVersionName>
    <autoTrackAppVersionCode>true</autoTrackAppVersionCode>
    <autoTrackAppPreInstalled>true</autoTrackAppPreInstalled>
    <autoTrackPlaystoreMail>false</autoTrackPlaystoreMail>
    <autoTrackPlaystoreGivenName>false</autoTrackPlaystoreGivenName>
    <autoTrackPlaystoreFamilyName>false</autoTrackPlaystoreFamilyName>
    <autoTrackApiLevel>true</autoTrackApiLevel>
    <autoTrackScreenOrientation>true</autoTrackScreenOrientation>
    <autoTrackConnectionType>true</autoTrackConnectionType>
    <autoTrackAdvertisementOptOut>true</autoTrackAdvertisementOptOut>
    <autoTrackRequestUrlStoreSize>true</autoTrackRequestUrlStoreSize>

    <enableRemoteConfiguration>false</enableRemoteConfiguration>
    <trackingConfigurationUrl></trackingConfigurationUrl>
    <resendOnStartEventTime>30</resendOnStartEventTime>
</webtrekkConfiguration>
```

Beachten Sie, dass die Konfigurationsdatei in Ihrem Android Modul unter dem Pfad „module/**res/raw/webtrekk_config.xml**“ zur Verfügung stehen muss.

2.2 Initialisieren des Tracking

Zur Nutzung der Tracking-Klassen innerhalb einer Quellcodedatei, müssen Sie diese importieren.

```
import com.webtrekk.webtrekksdk.Webtrekk;
```

Die Webtrekk Tracking Funktionalität wird über die Webtrekk Klasse zur Verfügung gestellt. Diese ist als Singleton implementiert und muss einmalig vor Nutzung initialisiert werden. Dies erfolgt am besten in der onCreate Methode Ihrer MainActivity.

```
private Webtrekk webtrekk;

@Override
protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        webtrekk = Webtrekk.getInstance();
        webtrekk.initWebtrekk(getApplication());
}
```

Insofern Sie die Konfigurationsdatei aus dem Kapitel 2.1 bereits hinterlegt und mit Ihrem Einstellungen ergänzt haben, steht Ihnen nun die Funktionalität des SDK zur Verfügung und Sie haben über das aktivieren des „autoTracked“ Elements in Ihrer Konfigurationsdatei bereits ein vollständiges Activity Tracking implementiert und senden nun erste Requests an Ihren Webtrekk Account.

Beim Verschieben der App in den Hintergrund oder beim direkten Schließen der App werden sämtliche noch nicht versendeten Request an Webtrekk übermittelt. Insofern eine Internetverbindung vorliegt.

2.3 Starten und Stoppen des Trackings

Im Gegensatz zu früheren Versionen des SDK, ist es nun nicht mehr notwendig, dass Sie zusätzlich die aufgerufene Activitiy übergeben müssen. Das Starten und Stoppen von Activities wird automatisch über das SDK ausgeführt und generiert, anhand des Modulpfades und des Activity-Namens, selbstständig eine Content-ID.

2.3.1 Seiten-Tracking

Wie auf einer Onlineplattform geht auch das App-Tracking davon aus, dass die Anwendung aus einzelnen Seiten besteht deren Nutzung erfasst werden soll. Das Seiten-Tracking ermöglicht Ihnen demnach die Übergabe von bestimmten App-Inhalten wie z.B. Seiten oder E-Commerce-Werten. Die Inhalte werden als Page-Impression gewertet und erscheinen in der Webtrekkoberfläche in der Seitenanalyse.

Wenn Sie das Element „autoTracked“ in Ihrem Konfigurations-XML deaktiviert haben, müssen Sie zum Erfassen der Activities das Messen der Activity manuell ausführen. Dies geschieht am besten in der „onStart()“-Methode Ihrer Activity mit dem Aufruf der „track()“-Methode aus dem Webtrekk SDK.

```
// manual tracking call
@Override
public void onStart(){
   super.onStart();
   webtrekk.track();
}
```

Bei aktiviertem „autoTracked“ wird das Seitentracking automatisch ausgeführt und sollte nicht manuell in der onStart Methode implementiert werden.

Ebenfalls können Sie zusätzliche Informationen mit einem Aufruf der „track()“-Methode übergeben, bzw. auch über die Konfigurationsdatei „webtrekk_config.xml“ globale Parameter (siehe Kapitel 3) und spezifische Parameter für einzelne Activities definieren (siehe Kapitel 4) werden. Um in den folgenden Beispielen ein „TrackingParameter“ Objekt zu erzeugen, müssen Sie diese Klasse ebenfalls importieren.

```
import com.webtrekk.webtrekksdk.TrackingParameter;
import com.webtrekk.webtrekksdk.TrackingParameter.Parameter;
```

Anschließend können Sie sich eine Variable vom Typ „TrackingParameter“ erzeugen, „“welcher Sie anschließend alle zu versendenden Daten zuweisen können.

```
// Example for a tracking call with a 'TrackingParameter' Object

[...]

private Webtrekk webtrekk;
private TrackingParameter tp;

@Override
protected void onCreate(Bundle savedInstanceState) {

   // get Webtrekk instance before
   [...]

   tp = new TrackingParameter();
   // tp.add(...)
}

@Override
public void onStart() {
   super.onStart();
   webtrekk.track(tp);
}

[...]
```

2.3.2 Contentgruppen (Seitenkategorien)

Mit Contentgruppen können Seiten zusammengefasst und somit Bereiche Ihrer App gebildet werden und direkt über die Tracking Library übergeben werden.

> Bei Fragen zu der Konfiguration von Contentgruppen und/oder deren Dateityp wenden Sie sich bitte an support@webtrekk.com oder Ihre(n) persönlichen Consultant.

```
// Tracking request with content groups:

private Webtrekk webtrekk;
private TrackingParameter tp;

@Override
protected void onCreate(Bundle savedInstanceState) {

   // Get Webtrekk Instance before
   [...]

   tp = new TrackingParameter();
   tp.add(Parameter.PAGE_CAT, "1", "de");
   tp.add(Parameter.PAGE_CAT, "2", "home");
}

@Override
public void onStart() {
   super.onStart();
   webtrekk.track(tp);
}
```

Contentgruppen-Parameter dürfen die Länge von 255 Zeichen nicht überschreiten. Alle weiteren Zeichen werden gelöscht.

> Contentgruppen werden einmalig einer Seite zugeordnet. Wird eine Seite zusammen mit einer Contentgruppe gemessen, werden alle folgenden Aufrufe dieser Seite auch dieser Contentgruppe zugeordnet.
> Bei der Übergabe der Contentgruppe mit der Tracking Library wird entsprechend nur der initiale Aufruf einer Seite (ContentID) beachtet. Folglich empfiehlt Webtrekk bei der Pixeleinbindung ContentIDs und Contentgruppen zeitgleich zu implementieren.

2.3.3 Seitenparameter (Eigene Parameter)

Mit "Eigenen Parametern" (paketabhängig) können Sie die Analysedaten mit Ihren App spezifischen Informationen bzw. Metriken anreichern.

Seitenparameter beziehen sich auf eine einzelne Seite und werden dieser direkt zugeordnet. Im Gegensatz zur Contentgruppe (siehe Kap. 2.3.2) muss dabei der Bezug zwischen der Seite und dem Seitenparameter nicht eindeutig sein. Damit können Sie den Aufruf einer Seite spezifizieren, z.B. durch Angabe der Variante oder eines Zahlenwertes.

```
// Tracking request with page parameter:

private Webtrekk webtrekk;
private TrackingParameter tp;

@Override
protected void onCreate(Bundle savedInstanceState) {

   // Get Webtrekk Instance before
   [...]

   tp.add(Parameter.PAGE, "1", "app");
   tp.add(Parameter.PAGE, "2", "android");
}

@Override
public void onStart() {
   super.onStart();
   webtrekk.track(tp);
}
```

2.3.4 Produkterfassung

Die folgenden Parameter dienen der detaillierten Messung von Produkten. Die Produkte können an Webtrekk übermittelt werden, wenn ein Produkt angesehen, in den Warenkorb gelegt oder wenn der Warenkorb gekauft wurde. Die Auflistung der gemessenen Produkte erfolgt im Webtrekk Tool unter „E-Commerce > Produkte“.

> Nutzen Sie keine Tausendertrenner in den Preisangaben. Dezimalstellen werden per Punkt oder Komma getrennt. Daten zu einer Bestellung werden beim „Aktionstracking“ **nicht** verarbeitet.

Produktname

Speichert die Produkte, die in den Warenkorb gelegt wurden. Wenn mehrere Produkte im Warenkorb liegen, werden diese jeweils mit einem Semikolon getrennt. Dieser Parameter muss zwingend ausgefüllt werden, wenn

Produkte gemessen werden sollen. Alle weiteren Parameter sind für die Produktmessung optional. Jedes Einzelprodukt darf nicht mehr als 110 Zeichen enthalten.

```
tp.add(Parameter.PRODUCT, "name-1;name 2");
```

Produktanzahl (optional)

Enthält die Produktanzahl. Wenn mehrere Produkte übertragen werden, werden diese jeweils mit einem Semikolon getrennt. Der Standardwert ist „1“.

```
tp.add(Parameter.PRODUCT_COUNT, "3;1");
```

Produktpreis (optional)

Enthält den Produktpreis („0“-Preise sind zulässig). Wenn Sie ein Produkt mehrfach übergeben (Produktanzahl größer 1), nutzen Sie den Gesamtpreis, nicht den Einzelpreis. Wenn mehrere Preise übertragen werden, werden diese jeweils mit einem Semikolon getrennt. Der Standardwert ist „0“.

```
tp.add(Parameter.PRODUCT_COST, "10.99;2999.95");
```

Währungscode (optional)

Enthält den Währungscode eines Produktes oder einer Bestellung, dabei muss der Wert nach ISO-Standard an das Webtrekk Pixel übergeben werden. Werden auf einer Seite mehrere Produkte übermittelt (z.B. auf der Bestellbestätigungsseite, wenn mehr als 1 Produkt gekauft wurde) gilt für alle Produkte nur 1 Währung. Dementsprechend muss der Wert auch nur einmal gesetzt werden.

```
tp.add(Parameter.CURRENCY, "EUR");
```

Hinweis: Die Übergabe der Währung dient lediglich der Währungsumrechnung. D.h. ggf. wird in die im Webtrekk-Frontend hinterlegte Währung (Konfiguration → Systemkonfiguration: Account) umgerechnet. Dort wird immer nur eine Währung ausgewiesen.

Warenkorbstatus (optional)

Enthält den Status des Warenkorbs. Wird ein Produkt angesehen (z.B. auf einer Produktdetailansicht), lautet der Status „view“. Dieser Status sollte immer dann gesetzt werden, wenn das Produkt in den Warenkorb gelegt werden kann.

Wenn ein Produkt in den Warenkorb gelegt wurde, lautet der Status „add“. Wurde der Warenkorb gekauft, wird der Status „conf“ übergeben. Wird bei Tracking eines Produktes kein Status übergeben, wird der Standardwert „view“, also eine Produktansicht, angenommen.

```
tp.add(Parameter.PRODUCT_STATUS, "view");
```

Gutscheinwert (optional)

Enthält den Wert eines Gutscheins. Nutzen Sie diesen Parameter, wenn der Kunde eine Bestellung mit einem Gutschein tätigt. Diesen Parameter können Sie nur bei Bestellungen übergeben.

```
tp.add(Parameter.VOUCHER_VALUE, "35.99");
```

Produktkategorien (optional)

Mit Produktkategorien können Produkte zusammengefasst werden. Der Bezug zwischen Produkt und Produktkategorie muss eindeutig sein. Beispielsweise ist es ist nicht möglich, dass das Produkt „Schuhe“ einmal der Produktkategorie „Damen“ und einmal der Produktkategorie „Sale“ zugeordnet wird. Solche nicht eindeutigen Beziehungen lassen sich über E-Commerce-Parameter (siehe Kap. 2.3.6) abbilden

Jede Produktkategorie darf nicht mehr als 110 Zeichen enthalten.

> Produktkategorien werden einmalig einem Produkt zugeordnet. Wird ein Produkt zusammen mit einer Kategorie gemessen, werden alle Produkte auch dieser Kategorie zugeordnet. Wenn vor dem Kauf eines Produktes zwingend der Produktstatus „view“ aufgerufen werden muss, reicht es daher, wenn Produktkategorien nur dort übergeben werden.

```
tp.add(Parameter.PRODUCT_CAT, "1", "men");
tp.add(Parameter.PRODUCT_CAT, "2", "shoes");
```

2.3.5 Messung von Bestellungen

Webtrekk bietet die Möglichkeit, Bestellungen zu messen. Neben der Bestellnummer wird dazu der Bestellwert übertragen. „0“-Werte sind zulässig. Der Unterschied zum Produkttracking ist, dass sich die Informationen nicht auf einzelne Produkte beziehen, sondern wie der Gesamtbestellwert Informationen zur Bestellung übermittelt übertragen werden. Der Gesamtbestellwert kann z.B. neben der Summe der gekauften Produkte auch Rabatte, Versandkosten und Verpackungskosten beinhalten.

> Nutzen Sie keine Tausendertrenner in den Preisangaben. Dezimalstellen werden per Punkt oder Komma getrennt.

Der Parameter „Gesamtbestellwert“ muss zwingend ausgefüllt werden, wenn Gesamtbestellwerte gemessen werden sollen.

Der Parameter „Bestellnummer“ (optional) enthält eine eineindeutige ID, die der Bestellung zugeordnet werden kann. Die Nutzung dieser Einstellung gewährleistet, dass keine Bestellungen doppelt gezählt werden.

```
tp.add(Parameter.ORDER_NUMBER, "M-12345");
tp.add(Parameter.ORDER_TOTAL, "2996.93");
tp.add(Parameter.VOUCHER_VALUE, "35.99");
```

2.3.6 E-Commerce Parameter

Mit "Eigenen Parametern" (paketabhängig) können Sie die Analysedaten mit Ihren webseitenspezifischen Informationen bzw. Metriken anreichern.

Bezug des Parameters

Mit E-Commerce-Parametern können weitere Produktinformationen (z.B. Größe, Farbe) übergeben werden. Bei mehreren Produkten muss die Anzahl der einzelnen Parameter-Werte mit der Anzahl der Produkte übereinstimmen. Die einzelnen Werte werden mit Semikolon getrennt.

E-Commerce-Parameter können aber auch verwendet werden, um Informationen zu einer Bestellung zu übergeben, z.B. Bezahlart, Versandart. In diesen Fällen ist das Messen von Bestellungen zwingend notwendig. Es reicht, diese Parameter je Bestellung einmal zu übergeben. Sie gelten gleichermaßen für alle Produkte im Warenkorb.

Der Bezug (Produkt oder Bestellung) wird bei der Konfiguration der Tracking Library gewählt. Ist „einzelner Wert“ gewählt bezieht sich der Parameter auf die Bestellung. Ist „mehrere Werte“ kann sich der Parameter sowohl auf Produkt als auch Bestellung beziehen.

> Hinweis: Da Webseitenziele in Webtrekk immer als E-Commerce Parameter erfasst werden müssen ist es auch möglich E-Commerce Parameter losgelöst von Bestellungen und Produkten zu übergeben.

```
tp.add(Parameter.ECOM, "1", "Levis");
tp.add(Parameter.ECOM, "2", "30");
tp.add(Parameter.ECOM, "3", "32");
```

2.3.7 Kampagnen

Das Kampagnen-Tracking wird im Webtrekk Tool konfiguriert (Konfiguration > Kampagnenkonfiguration). Ohne diese Konfiguration werden keine Kampagneninformationen, wie z.B. Kampagnenklicks, erfasst. Als Kampagnenklick kann unter anderem der Aufruf bestimmter Seiten in einer App oder die Erfassung definierter Verweise gemessen werden.

Es besteht die Möglichkeit, selbst eine Kampagnen-ID im Konfigurationsteil zu setzen. Eine Kampagnen-ID besteht aus einem Mediacode-Namen und dem dazugehörigen Wert, getrennt durch ein „%3D“.

Ebenso können Kampagnen mittels Kampagnenparameter um Zusatzinformationen ergänzt werden.

```
tp.add(Parameter.ADVERTISEMENT, "mc%3Dnewsletter_2016_1");
```

Kampagnenparameter

Mit "Eigenen Parametern" können Sie die Analysedaten mit Ihren App seitenspezifischen Informationen bzw. Metriken anreichern.

Kampagnenparameter beziehen sich immer auf ein Werbemittel (kleinste in Webtrekk vorhandene Untereinheit einer Kampagne).

```
tp.add(Parameter.AD, "1", "personalized");
```

Webtrekk bietet die Möglichkeit, Google Kampagnen automatisch zu erfassen, wenn über diese die App aus dem App-Store geladen und installiert wurde.

Anhand des Referrer-Parameters kann nachvollzogen werden, über welche Wege der Nutzer im Play Store zur App gekommen ist.

Zuerst muss der BroadcastReceiver der Library in das Manifest der Applikation eingetragen werden. Dazu öffnet man die XML Ansicht des Manifests und fügt folgenden Code innerhalb des <application> Tags ein:

```
<receiver
   android:name="com.webtrekk.webtrekksdk.ReferrerReceiver"
   android:exported="true">
   <intent-filter>
      <action android:name="com.android.vending.INSTALL_REFERRER" />
   </intent-filter>
</receiver>
```

Der Aufbau von Links zum Play Store samt Informationen zum Kampagnen-Tracking ist auf der folgenden Website dokumentiert: https://developers.google.com/analytics/devguides/collection/android/v2/campaigns. Bis auf „gclid“ werden alle dort angegeben Parameter unterstützt.

Die von uns erfasst Kampagne setzt sich dann wie folgt zusammen:

- Media Code (wt_mc): utm_source.utm_medium.utm_content.utm_campaign
- Kampagnen Keyword (wt_kw): utm_term

```
https://play.google.com/store/apps/details?id=com.example.app
&referrer=utm_source%3Dgoogle
%26utm_medium%3Dcpc
%26utm_term%3Drunning%252Bshoes
%26utm_content%3DdisplayAd1
%26utm_campaign%3Dshoe%252Bcampaign
```

Der Kampagnencode muss, wie im Beispiel oben, URL-kodiert im „referrer“-Parameter im Link auf den Playstore enthalten sein.

2.3.8 Sessionparameter

Sessionparameter beziehen sich immer auf eine Session, also einen Visit. Wird der Wert für den Parameter innerhalb eines Visits mehrmals übertragen, wird jeweils nur der zuletzt übermittelte Wert ausgewertet.

Zum Beispiel könnte der Status, ob ein User während des Visits eingeloggt war, übergeben werden. Standardmäßig würde jeder Besuch zu Beginn in einem Sessionparameter als „nicht eingeloggt“ gekennzeichnet. Ein-Login wird an den gleichen Parameter übergeben und überschreibt damit den ersten Wert.

> Im Unterschied zu einem Seitenparameter kann nicht ausgewertet werden, auf welcher Seite ein Sessionparameter gesetzt wurde. Zudem belässt ein Seitenparameter jeden Wert auswertbar, der innerhalb des Visits gesetzt wurde.

```
tp.add(Parameter.SESSION, "1", "logged.in");
```

2.3.9 Interne Suche

Analysieren Sie die Suchbegriffe, die Besucher in Ihrer App eingeben, indem Sie diese im Tracking mitgeben. Tragen Sie dynamisch den genutzten Suchbegriff in den Konfigurationsparameter ein.

```
tp.add(Parameter.INTERN_SEARCH, "searchterm");
```

2.3.10 Eigene Besucher-IDs

Um die Besuchererkennung zu verbessern, können Sie anstatt des Webtrekk Ever-ID, eigene Besucher-IDs verwenden.

Für die Nutzung eigener Besucher-IDs geben Sie der Tracking Library einen eindeutigen Identifikator aus Ihrem System mit. Sollten Sie keine eindeutigen Besucher-IDs in Ihrer App einsetzen, ist alternativ die Emailadresse des Besuchers als eindeutiger Identifikator denkbar. In diesem Fall sollten Sie aus datenschutzrechtlichen Gründen die Emailadresse unlesbar machen (z.B. mit dem MD5 Hash).

```
tp.add(Parameter.CUSTOMER_ID, "372d1a04d003eebc09e17330d5d3117c");
```

Mit den optionalen Besucherkategorien können Sie zusätzlich den Besucher kategorisieren. Diese URM Kategorien müssen zuvor im Tool angelegt werden. Im unten gezeigten Beispiel wird dem Besucher, der Familienstand zugewiesen.

```
tp.add(Parameter.USER_CAT, "1", "single");
```

2.4 Aktions-Tracking

Mit dem Aktions-Tracking können Sie Ereignisse in der App wie zum Beispiel Klicks auf Buttons oder Auswählen von Checkboxen verfolgen.

```
TrackingParameter buttonParameter = new TrackingParameter();

buttonParameter
   .add(Parameter.ACTION_NAME, "orderButton")
   .add(Parameter.CURRENCY, "EUR")
   .add(Parameter.PRODUCT, "product-a")
   .add(Parameter.PRODUCT_COUNT, "1")
   .add(Parameter.PRODUCT_STATUS, "add");

webtrekk.track(buttonParameter);
```

Analog zum Seiten-Tracking können Sie bei jedem Request weitere Parameter wie z.B. Platzierung oder Farbe eine Buttons anhängen.

```
tp.add(Parameter.ACTION, "1", "green ");
tp.add(Parameter.ACTION, "2", "top");
```

2.5 Media Tracking

Webtrekk bietet Ihnen die Möglichkeit, die Mediennutzung in Ihrer App zu erfassen. Das Tracking erfolgt analog zu den bisher gezeigten Beispielen.

Um eine Media Datei zu tracken, richten Sie zunächst die entsprechenden TrackingParameter ein, dort können Sie den Dateinamen, die Länge, die Bandbreite und weitere Details konfigurieren.

Im Anschluss daran, können Sie in den entsprechenden Aktionsmethoden wie Play, Pause, Stop, Suche jeweils einen Tracking Request versenden. Das folgende Beispiel zeigt eine Erweiterung der Android VideoPlayer Klasse welche die Methoden der Basis Klasse um das Tracking ergänzt.

```
public class TrackedVideoView extends VideoView {

private Webtrekk webtrekk;
TrackingParameter tp;

    public TrackedVideoView(Context context) {
        super(context);
        initMediaFile();
    }

    public TrackedVideoView(Context context, AttributeSet attrs) {
        super(context, attrs);
        initMediaFile();
    }

    public TrackedVideoView(Context context,
AttributeSet attrs, int defStyle) {

        super(context, attrs, defStyle);
        initMediaFile();
    }

    public void initMediaFile() {
        tp = new TrackingParameter();

        tp.add(Parameter.MEDIA_FILE,
                getResources().getResourceEntryName(R.raw.marv3));
        tp.add(Parameter.MEDIA_LENGTH,
                String.valueOf(getDuration()));
        tp.add(Parameter.MEDIA_POS,
                String.valueOf(getCurrentPosition()));
        tp.add(Parameter.MEDIA_CAT, "1", "mp3");
        tp.add(Parameter.MEDIA_CAT, "1", "example");
    }

    @Override
    public void pause() {
        super.pause();
        tp.add(Parameter.MEDIA_ACTION, "pause");
        tp.add(Parameter.MEDIA_POS,
                String.valueOf(getCurrentPosition()));
        webtrekk.track(tp);
    }

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

        tp.add(Parameter.MEDIA_ACTION, "start");
        tp.add(Parameter.MEDIA_POS,
                String.valueOf(getCurrentPosition()));

        webtrekk.track(tp);
    }

[...]
```

```
[...]

    @Override
    public void seekTo(int msec) {
        super.seekTo(msec);

        tp.add(Parameter.MEDIA_ACTION, "seek");
        tp.add(Parameter.MEDIA_POS, String.valueOf(msec));

        webtrekk.track(tp);
    }

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

        tp.add(Parameter.MEDIA_POS,
                String.valueOf(getCurrentPosition()));
        tp.add(Parameter.MEDIA_ACTION, "stop");

        webtrekk.track(tp);
    }

    public Webtrekk getWebtrekk() {
        return webtrekk;
    }

    public void setWebtrekk(Webtrekk webtrekk) {
        this.webtrekk = webtrekk;
    }
}
```

Eine Media-Session wird geöffnet, sobald das Video für den Nutzer sichtbar ist. So wird auch die Anzeige des Videos (auch wenn es noch nicht abgespielt wird) gemessen.

Jede Media-Session wird beendet, sobald die Aktion „stop“ erfasst wird. Sollen danach noch weitere Ereignisse folgen, muss vorher eine neue Media-Session geöffnet werden.

2.5.1 Play

Das Play-Ereignis muss gesendet werden, wenn der Nutzer die Wiedergabe des Videos startet oder aber die Wiedergabe nach einem Pause-Ereignis fortsetzt.

```
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "init");
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "play");
```

2.5.2 Position

Das Position-Ereignis sollte während der Wiedergabe des Videos im Abstand von 30 Sekunden gesendet werden. So kann die Ansicht des Videos akkurat gemessen werden.

```
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "pos");
```

2.5.3 Pause

Das Pause-Ereignis muss gesendet werden, wenn der Nutzer die Wiedergabe des Videos pausiert.

```
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "pause");
```

2.5.4 Seek

Das Seek-Ereignis muss einmalig gesendet werden, wenn der Nutzer beginnt, die Position des Videos zu verändern. Wird der Schieberegler des Medien-Players wieder losgelassen, muss eine weitere Aktion zum Beenden des Spulvorganges gesendet werden. Diese zweite Aktion ist abhängig vom Modus, in dem sich der Medien-Player befindet. Während der Wiedergabe wird als zweite Aktion ein „play“ gesendet, befindet sich der Player gerade im Pause-Modus, wird das Ende des Spulvorganges durch eine „pause“ Aktion bestätigt.

```
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "seek");
```

2.5.5 Stop

Das Stop-Ereignis muss gesendet werden, wenn der Nutzer die Wiedergabe des Videos stoppt, das Ende des Videos erreicht wurde oder das Video entfernt wird (z.B. wenn die Activity geschlossen oder eine andere geöffnet wird).

```
MediaActivity.this.tp.add(Parameter.MEDIA_ACTION, "stop");
```

Das Stop-Ereignis beendet die Media-Session. Zum messen weiterer Ereignisse (z.B. einer erneuten Wiedergabe), muss die Media-Session neu erstellt wird.

3 Globale Tracking Parameter

Globale Tracking Parameter stehen über Activities hinweg zur Verfügung und sind somit privilegiert dafür konstante Werte oder gewünschte Informationen aus den StandardCustomParametern für jede einzelne Activity zu versenden. Dazu steht Ihnen innerhalb Ihrer App das GlobalTrackingParameter Objekt zur Verfügung. In diese Instanz von TrackingParameter können Sie globale Werte setzen welche mit jedem Request versendet werden:

```
// 'apiLevel' is a keyword of custom tracking parameter
// each request contains the api level of the device
// in session parameter 6
webtrekk.getGlobalTrackingParameter().add(
   Parameter.SESSION,
   "6",
    webtrekk.getCustomParameter().get("apiLevel")
);
```

Alternativ können Sie in Ihrer Konfigurationsdatei dasselbe Szenario abbilden. Beachten Sie das im Sourcecode gesetzte GlobaleTrackingParameter, von denen in Ihrer Konfigurationsdatei überschrieben werden.

```
<globalTrackingParameter>
   <sessionParameter>
      <parameter id="6" key="apiLevel" value=""></parameter>
   </sessionParameter>
</globalTrackingParameter>
```

Sie können jeden Parameter, welcher Ihnen in dem TrackingParameter Objekt zur Verfügung steht, als Globalen Parameter setzen und somit mit jedem Request versenden.

> Requests mit Aktionsnamen sind hiervon nicht betroffen. Mehr Informationen zum Aktionstracking finden Sie im Kapitel zum Aktionstracking.

Im Folgenden finden Sie eine Übersicht zur Konfiguration Globaler Tracking Parameter für Ihre Konfigurationsdatei:

```
<globalTrackingParameter>

   <!-- define global tracking parameter which are send with every
request, the key has to match a valid parameter name!
   entries made here are available as default parameters in the
trackingparameter instance -->
   <parameter id="PRODUCT">product-a</parameter>
   <parameter id="PRODUCT_COST">0.99</parameter>

   <!-- define the global page parameter, the key is the index -->
   <pageParameter>
      <parameter id="1">pageparam1</parameter>
      <parameter id="2">pageparam2</parameter>
      <parameter id="3">pageparam3</parameter>
   </pageParameter>

   <sessionParameter>
      <parameter id="1">sessionparam1</parameter>
   </sessionParameter>

   <ecomParameter>
      <parameter id="1">ecomparam1</parameter>
   </ecomParameter>

   <userCategories>
      <parameter id="1">usercategory1</parameter>
   </userCategories>

   <pageCategories>
      <parameter id="1">pagecategory1</parameter>
   </pageCategories>

   <adParameter>
      <parameter id="1">adparam1</parameter>
   </adParameter>

   <actionParameter>
      <parameter id="1">actionparam1</parameter>
   </actionParameter>

   <productCategories>
      <parameter id="1">productcategory1</parameter>
   </productCategories>

   <mediaCategories>
      <parameter id="1">mediacategory1</parameter>
   </mediaCategories>

</globalTrackingParameter>
```

4 Activity Parameter

Im Gegensatz zu den Globalen Tracking Parametern, stehen Ihnen die Activity Parameter nur im Scope der Activity zur Verfügung. Alle Parameter welche im Sourcecode Ihrer App innerhalb einer Activty einem „TrackingParameter“ Objekt zugewiesen werden, sind automatisch Activity Parameter.

Dadurch das Activity Parameter einen näheren Bezug zur Activity haben, überschreiben Sie die Konfiguration für Globale Tracking Parameter. Dies ist für, im Sourcecode und auch in der Konfigurationsdatei gesetzte, Globale Tracking Parameter der Fall. Ebenfalls werden, wie auch bei den Globalen Tracking Parametern, im Sourcecode definierten Konfigurationen für einen jeden Activity Parametern, von der Konfiguration aus der Konfigurationsdatei überschrieben.

Activity Parameter werden innerhalb des Sourcecodes, wie in den Beispielen von Kapitel 2.4 gesetzt. Im Folgenden nochmal ein kurzes Beispiel:

```
// Example for a tracking call with an object TrackingParameter

private Webtrekk webtrekk;
private TrackingParameter tp;

[...]

@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);

   webtrekk = Webtrekk.getInstance();

   tp = new TrackingParameter();
   tp.add(Parameter.PAGE_CAT, "1", "de");
   tp.add(Parameter.PAGE_CAT, "2", "home");
}

@Override
public void onStart() {
   super.onStart();
   // no track call, because autoTracked is enabled
   // webtrekk.track(tp);
}

[...]
```

Alternativ können Sie in Ihrer Konfigurationsdatei dasselbe Szenario abbilden:

```
<activity>
   <classname type="text">myapp.test.MainActivity</classname>
   <mappingname type="text">Landingpage</mappingname>
   <autoTracked>true</autoTracked>

   <!--activity tracking parameter -->
   <activityTrackingParameter>
      <pageCategories>
         <parameter id="1">de</parameter>
         <parameter id="2">home</parameter>
      </pageCategories>
   </activityTrackingParameter>
</activity>
```

Requests mit Aktionsnamen sind hiervon nicht betroffen. Mehr Informationen zum Aktionstracking finden Sie im Kapitel zum Aktionstracking.

Für jede Activity welche über die Konfigurationsdatei angepasst/erweitert werden soll, wird ein „<activity></activity>“ Element in der Konfigurationsdatei definiert. Für dieses Element gibt es über folgende Konfigurationselemente, weitere Einstellungsmöglichkeiten:

- **<classname></classname>**: Enthält den Activitynamen, für welchen die nachfolgenden Einstellungen gültig sind
- **<mappingname></mappingname>**: mit diesem Element können Sie eine alternative Content ID für diese Activity angeben
- **<autoTracked></autoTracked>**: Aktiviert/deaktiviert separat für diese Activity das automatische tracken der Activity

Im Folgenden finden Sie eine Übersicht zur Konfiguration Globaler Tracking Parameter für Ihre Konfigurationsdatei:

```
<activity>
   <classname type="text">myapp.test.MainActivity</classname>
   <mappingname type="text">Landingpage</mappingname>
   <autoTracked>true</autoTracked>

   <!--activity tracking parameter -->
   <activityTrackingParameter>
      <pageParameter>
         <parameter id="1">pageParam1</parameter>
         <parameter id="2">pageParam2</parameter>
         <parameter id="3">pageParam3</parameter>
      </pageParameter>
      <sessionParameter>
         <parameter id="1">sessionParam1</parameter>
      </sessionParameter>
      <ecomParameter>
         <parameter id="1">ecomParam1</parameter>
      </ecomParameter>
      <userCategories>
         <parameter id="1">userCat1</parameter>
      </userCategories>
      <pageCategories>
         <parameter id="1">pageCat1</parameter>
      </pageCategories>
      <adParameter>
         <parameter id="1">adParam1</parameter>
      </adParameter>
      <actionParameter>
         <parameter id="1">actionParam1</parameter>
      </actionParameter>
      <productCategories>
         <parameter id="1">productCat1</parameter>
      </productCategories>
      <mediaCategories>
         <parameter id="1">mediaCat1</parameter>
      </mediaCategories>

   </activityTrackingParameter>
</activity>
```

5 Custom Parameter / Placeholder

Über die Custom Parametern haben Sie Zugriff über alle im Konfigurations-XML aktivierten Eigenschaften siehe

Tracking-Konfiguration. Zusätzlich können Sie in die Custom Parameter selbstdefinierte Keys schreiben und deren Wert im Scope der Activity in freiwählbaren Parametern verwenden.

```
// set key 'example_product_name'with value to custom parameter
webtrekk.getCustomParameter().put("my_product_name", "Product A");

// set the value of custom parameter 'example_product_name' to the
// Webtrekk Product name
tp.add(
   TrackingParameter.Parameter.PRODUCT,
   webtrekk.getCustomParameter().get("my_product_name")
);
```

Wenn Sie definierte Custom Parameter über Ihre Konfigurationsdatei einem Tracking Parameter zuweisen möchten, nutzen Sie das „key"-Attribut eines jeden Elements um den Wert des Custom Parameters mit dem zu trackenden Parameter zu versenden:

```
// set the value of custom parameter 'example_product_name' to the
// Webtrekk Product name by using the 'key' attribute
<parameter id="PRODUCT" key="my_product_name"></parameter>
```

Achten Sie darauf, dass Custom Parameter initialisiert sind, bevor diese genutzt werden.

5.1 Standard Custom Parameter

Webtrekk bietet die Möglichkeiten diverse vordefinierte Eigenschaften aus der Tracking Library auszulesen und in einen freiwählbaren Parameter zu versenden. Im Folgenden einen Übersicht der Standardparameter:

Element (Key)	Beschreibung	Werte
appPreinstalled	Enthält ob die App bereits auf dem Gerät vorinstalliert war.	true | false
appVersion	Enthält die Android App Version.	bspw. 2.1
appUpdated	Enthält die Information, ob die App geupdatet wurde.	0 | 1
apiLevel	Enthält das Android API-Level.	bspw. 19
appVersionCode	Immer Wenn Sie eine neue Version ihrer App im Playstore veröffentlichen erhält diese eine eindeutige Integer Zahl zur Identifizierung. Diese ist aufsteigend und ermöglicht das Erkennen von Updates.	bspw. 3
screenOrientation	Enthält die aktuelle Ausrichtung des Geräts.	portrait | landscape
connectionType	Enthält den aktuellen Verbindungstyp	3G | WiFi | Offline
requestUrlStoreSize	Enthält die aktuelle Anzahl an zwischengespeicherten Requests und bietet Ihnen somit die Möglichkeit zu ermitteln, ob die in der Konfigurationsdatei hinterlegt Größe für den URL-Store optimal bemessen ist oder dieser bspw. zu klein konfiguriert ist	bspw. 173
playstoreMail	Enthält die mit dem ersten Google Konto des Gerätes assoziierte Email-Adresse.	Bspw.: john.doe@domain.org
playstoreGivenname	Enthält den Vornamen des Benutzers des ersten Google Kontos eines Gerätes.	Bspw.: John
playstoreFamilyname	Enthält den Nachnamen des Benutzers des ersten Google Kontos eines Gerätes.	Bspw.: Doe
advertiserId	Die Advertiser ID ist eine eindeutige Benutzerkennung die Google über den Playstore an jeden Benutzer vergibt. Diese ermöglicht es Benutzer wiederzuerkennen auch über verschiedene Geräte hinweg, solange sie sich mit dem selben Google Account anmelden.	Bspw.: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
advertisingOptOut	Gibt dem Anwender der App die Möglichkeit keine Werbung zu sehen.	true | false

6 Aktionstracking

Das Aktionstracking ermöglicht Ihnen das zusätzliche Messen von Interaktionen innerhalb einer Activity. Beispielsweise können Sie das klicken auf eine Schaltfläche messen, indem Sie beim Eintreten des gewünschten Zustands der Schaltfläche, einen Aktionsrequest übergeben.

Was ist ein Aktionsrequest? Jeder Request der einen „ACTION_NAME“ übergeben bekommt und somit im Trackrequest den „ct-Parameter“ befüllt ist ein Aktionsrequest.

Das messen von Aktionen kann nur manuell erfolgen und muss somit im Sourcecode Ihrer App erfolgen. Anders als beim normalen Erfassen von Daten, ist es beim Aktionstracking nur möglich die Daten zu erfassen, welche auch der Methode „track“ als Parameter übergeben werden. Das bedeutet, dass mit einem Aktionsrequest keine Globalen Tracking Parameter versendet werden, vorausgesetzt sie werden nicht explizit an die Methode „track“ übergeben.

```
// track an action by adding an 'ACTION_NAME'
// only the given parameters will set to the tracking request -
// no global or activity parameters
public void onAddToBasketClicked(View view) {
TrackingParameter buttonParameter = new TrackingParameter();

   buttonParameter
      .add(Parameter.ACTION_NAME, "addToBasket")
      .add(Parameter.CURRENCY, "EUR")
      .add(Parameter.PRODUCT, "product-a")
      .add(Parameter.PRODUCT_COUNT, "1")
      .add(Parameter.PRODUCT_STATUS, "add");

   webtrekk.track(buttonParameter);
}
```

7 Tracking Parameter Objekt

Über die Instanz eines TrackingParameter Objekts haben Sie die Möglichkeit, eigene Parameter zu setzen und auch Standardparameter übersichtlich und einheitlich zu übergeben:

Key			URL-Parameter
ACTION_NAME	Aktionsname		ct
VOUCHER_VALUE	Gutscheinwert		cb563
ORDER_TOTAL	Bestellwert		ov
ORDER_NUMBER	Bestellnummer		oi
PRODUCT	Produktname/-n		ba
PRODUCT_COST	Produktkosten		co
CURRENCY	Währung		cr
PRODUCT_COUNT	Produktanzahl		qn
PRODUCT_STATUS	Produktstatus		st
CUSTOMER_ID	Kunden ID		cd
EMAIL	E-Mail des Nutzers		uc700
EMAIL_RID			uc701
NEWSLETTER			uc702
GNAME	Vorname des Nutzers		uc703
SNAME	Nachname des Nutzers		uc704
PHONE	Telefon		uc705
GENDER	Geschlecht		uc705
BIRTHDAY	Geburtsdatum		uc707
CITY	Stadt		uc708
COUNTRY	Land		uc709
ZIP	Postleitzahl		uc710
STREET	Straße		uc711
STREETNUMBER	Hausnummer		uc712
INTERN_SEARCH	Suchbegriff		is
ACTION	Aktionsparameter		z.B. ck1
AD	Kampagnenparameter		z.B. cc1
ECOM	E-commerceparameter		z.B. cb1
PAGE	Seitenparameter		z.B. cp1
SESSION	Sessionparameter		z.B. cs1
MEDIA_CAT	Medienkategorien		z.B. mg1
PAGE_CAT	Seitenkategorien		z.B. cg1
PRODUCT_CAT	Produktkategorien		z.B. ca1
USER_CAT	Nutzerkategorien		z.B. uc1

8 Webtrekk Ever-ID

Die Webtrekk Ever-ID wird verwendet um Nutzer zu erkennen. Auf einer Standardwebseite wird diese automatisch gesetzt und übermittelt.

Da die technischen Möglichkeiten zu Nutzererkennung auf mobilen Geräten eingeschränkt sind, haben Sie die Möglichkeit diese ID manuell auszulesen und zu übermitteln, wenn Ihre Nutzer z.B. von der App in den mobilen Browser wechseln. Somit behält ein solcher Nutzer während dem Besuch der Webseite die Ever-ID aus der App. Der Besuch kann somit übergreifend gemessen werden.

Zum Auslesen der Ever-ID können Sie unten beschriebene Methode verwenden.

```
String eid = webtrekk.getEverId();
```

Das Übermitteln an den Browser erfolgt durch das Anhängen zweier Parameter an den Ziel-Link (URL).

- „wt_eid“: die Ever-ID
- „wt_t“: aktueller Unix-Timestamp in Millisekunden. Dieser ist notwendig, dass die Ever-ID in der URL eine maximale Gültigkeit von 15 Minuten hat. Somit wird die Wahrscheinlichkeit, dass ein solcher Link z.B. gepostet wird und dieser so häufig mit demselben IDs gewertet wird, minimiert (sonst würden alle Besucher dieselbe EverId erhalten und so als 1 Besucher zählen!).

```
// Example for an URL with Ever-ID and timestamp:
http://new.domain.com/page?wt_eid=2135817235100536325&wt_t=135841437
8580
```

Beim Klick auf den Link öffnet sich der Webbrowser und der Nutzer gelangt auf die neue Seite. Dort erhält er die gleiche Ever-ID, die er auch in der App besessen hat.

Zusätzlich dazu können Sie noch einen eigenen Referrer mitgeben, damit der Einstig in Ihre Webseite nicht als Direkt ausgewiesen wird. Zum Simulieren eines eigenen Referrers hängen Sie zusätzlich den URL-Parameter „wt_ref“ an die Link-URL. Auch hier wird wieder der Parameter „wt_t“ verwendet, der eine maximale Gültigkeit von 15 Minuten hat.

```
// Example URL with Ever-ID, timestamp and referrer:
http://new.domain.com/start.html?wt_eid=2135817235100536325&wt_ref=h
ttp%3A%2F%2Fwww.webtrekk.com%2Fen%2Fhome.html&wt_t=1358414378580
```

Bitte achten Sie darauf, dass die Referrer-URL kodiert werden muss.

Hinweis: Durch falsche Handhabung dieses Features kann es passieren, dass viele Nutzer die gleiche Ever-ID erhalten. Dies würde extreme Auswirkungen auf Ihre Datenqualität mit sich ziehen. Bitte achten Sie bei Einsatz auf die korrekte Verwendung!

Dieses Feature setzt voraus, dass Sie auf der Webseite die **Pixelversion 3.2.3** oder höher nutzen.

9 Opt-Out Funktionalität

Nach §15 des Telemediengesetzes können Webseitenbesucher der Datenspeicherung Ihrer anonymisiert erfassten Besucherdaten widersprechen, so dass sie in Zukunft nicht mehr erfasst werden. Um diesen Widerspruch zu gewährleisten, muss die Methode „setOptedOut“ genutzt werden.

```
// exclude user from tracking
webtrekk.setOptout(true);
```

Mit dem Aufruf „optedOut“ können Sie prüfen, ob der Nutzer gemessen werden möchte oder nicht.

```
// get information if user has optout status
boolean optOut = webtrekk.isOptedOut();
```

10 Externe Trackingkonfiguration

Um bei einer fehlerhaften Trackingkonfiguration oder einer Anpassung dieser nicht ein vollständiges Release in den Google Playstore einspielen zu müssen, haben Sie die Möglichkeit einen Pfad in Ihrem Konfigurations-XML zu hinterlegen, von welchem bei jedem Start der Applikation versucht wird ein neueres Konfigurationsskript zu laden. Diese nachgeladene Datei überschreibt die bisherige Konfiguration vollständig.

Um das nachladen einer externen Konfigurationsdatei zu aktivieren, müssen Sie lediglich die folgenden 2 Elemente in Ihrer Konfigurationsdatei konfigurieren:

```
<enableRemoteConfiguration>true</enableRemoteConfiguration>
<trackingConfigurationUrl>https://app.domain.tld/tracking_config.xml
</trackingConfigurationUrl>
```

Wenn Sie eine externe Konfigurationsdatei nachladen möchten, muss diese valide sein, d.h. das alle unter Kapitel 2.1 aufgeführten Pflichtkonfigurationen ebenfalls in der externen Konfigurationsdatei hinterlegen sein müssen, da andernfalls die nachgeladene Konfigurationsdatei verworfen wird und die bereits vorhandene Konfigurationsdatei genutzt wird.

> Zusätzlich muss die Versionsnummer in der nachzuladenden Konfigurationsdatei einen höheren Wert als in der aktuellen Konfigurationsdatei haben. Da die Datei sonst nicht als Update interpretiert werden kann.

Beachten Sie bitte die folgenden Konfigurationspunkte zum Nachladen einer externen Konfigurationsdatei:

1. „enableRemoteConfiguration“ ist in der aktuellen Konfiguration aktiviert
2. Die unter „trackingConfigurationUrl“ in der aktuellen Konfiguration hinterlegte Datei ist verfügbar; achten Sie bitte auch auf das Protokoll welches Sie verwenden
3. Neue Konfigurationsdatei hat die richtigen Angaben unter:
    a. TrackID
    b. Trackdomain
    c. „sampling“, „sendDelay“, „maxRequests“, „autoTracked“
    d. „enableRemoteConfiguration“
    e. „trackingConfigurationUrl“
4. Versionsnummer in neuerer Konfigurationsdatei erhöhen

> Wenn Ihre Applikation bereits im Playstore erhältlich ist, ist es nicht möglich das nachladen einer externen Konfigurationsdatei, ohne ein Update Ihrer Applikation im Playstore, zu aktivieren.

11 Kontakt

Wenn Sie Fragen zur Einrichtung haben sollten, stehen wir Ihnen selbstverständlich zur Verfügung. Für priorisierten Support oder umfassendere Beratung bietet Webtrekk Support- und Consultingpakete an. Sprechen Sie uns an, wir unterbreiten Ihnen gerne ein individuelles Angebot.

Webtrekk GmbH
Robert-Koch-Platz 4
10115 Berlin

fon 030 - 755 415 - 0
fax 030 - 755 415 - 100
support@webtrekk.com

www.webtrekk.com