Dezember 2012

# Facebook Tracking & Monitoring

# - Teil 2 -

Exklusiver Auszug aus dem Buch Facebook Fanpages Plus von Tim Sebastian


allfacebook.de
Der inoffizielle Facebook Blog

*Der erste Teil dieses Auszugs aus dem Buch Facebook Fanpages Plus von Tim Sebastian ist bereits am 10. Dezember 2012 auf AllFacebook.de erschienen und dort verfügbar.*

# 2. Einsatz von Webreporting-Tools

Obwohl die Informationen, die Sie auf Facebook Insights erhalten, interessante Aufschlüsse über Ihre Fans, deren Altersstruktur und Interaktionsfreudigkeit geben, können Sie mit diesen Daten nicht ein bisheriges Webreporting-Tool ersetzen.

Die folgenden Abschnitte beschreiben, welche Daten Sie auf Ihrer Fanpage und Ihrer Website über Facebook erhalten und wie Sie diese Daten in Ihr Webreporting-Tool einbinden und darstellen können.

Die Beispiele in diesem Kapitel werden mit Hilfe des frei verfügbaren Webreporting-Tools Google Analytics vorgenommen, sollten aber in jedem vernünftigen Webreporting-Tool ebenfalls umsetzbar sein.

## 2.1. Kurze Einführung in Google Analytics

Google Analytics ist ein von Google angebotenes Webreporting-Tool, das Websitebetreibern die Möglichkeit bietet, Daten über Nutzeranzahl und Verhalten auf den eigenen Webpages zu messen und auszuwerten.

Dazu muss ähnlich dem JavaScript-SDK von Facebook ein JavaScript-Code auf der Webpage eingebunden werden. Dieser JavaScript-Code bildet eine Schnittstelle zwischen der Website und dem Tracking-Server und kann über diverse Methoden aufgefordert werden, Daten an den Tracking-Server zu schicken.

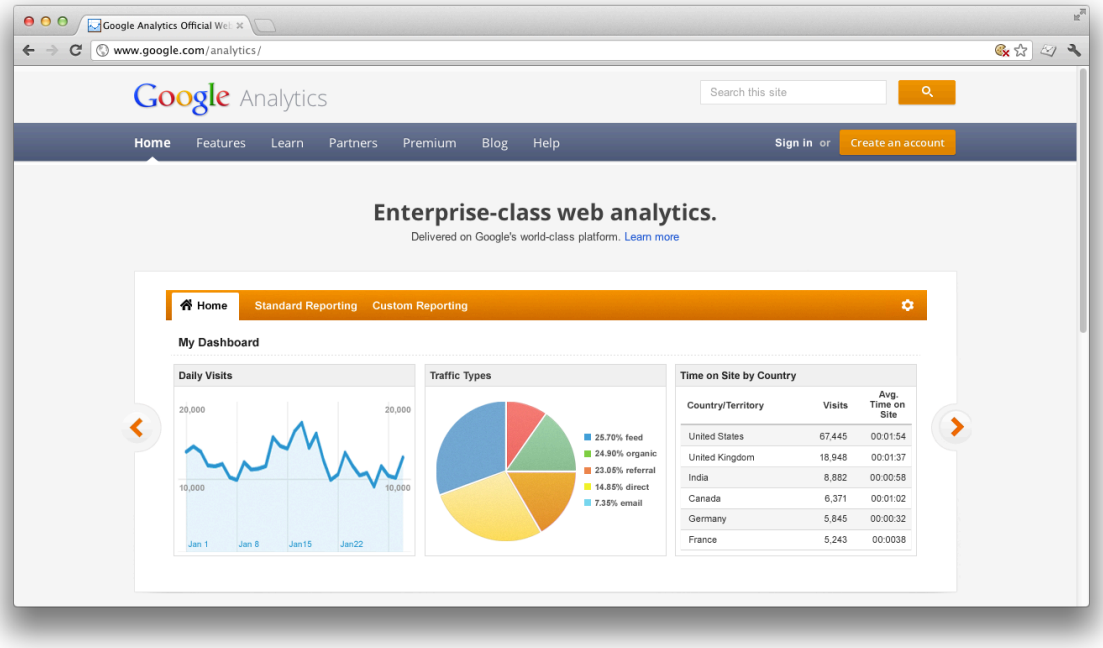



Abbildung 6.10: Google Analytics – Kostenfreies Webreporting-Tool

## Standard-Tracking-Snippet für Google Analytics

Gerade wenn man sich Google-Analytics-Implementierungen ansieht, bleibt dieser Funktionsumfang oft ungenutzt, stattdessen wird einfach der Standard-Trackingcode auf die Webpages kopiert. Dabei zeigt ein kurzer Blick auf genau diesen Standard-Trackingcode bereits, dass dieser aus einer Reihe von Aufrufen besteht.

```
<script type="text/javascript">
  var _gaq = _gaq || [];
  _gaq.push(['_setAccount', 'UA-XXXXX-X']);
  _gaq.push (['_gat._anonymizeIp']);
  _gaq.push(['_trackPageview']);

  (function() {
    var ga = document.createElement('script'); ga.type = 'text/
javascript'; ga.async = true;
    ga.src = ('https:' == document.location.protocol ? 'https://ssl' :
'http://www') + '.google-analytics.com/ga.js';
```

```
    var s = document.getElementsByTagName('script')[0];
s.parentNode.insertBefore(ga, s);
  })();
</script>
```

Listing 6.1: Standard-Google-Analytics-Tracking-Snippet

Erklärung des Codes:

```
var _gaq = _gaq || [];
```

Das `_gaq`-Objekt wird initialisiert, indem geprüft wird, ob es bereits im globalen Namespace existiert. Ist dies nicht der Fall, wird der `gaq`-Variablen ein leeres Array zugewiesen. Auf diese Weise können bereits Methoden in die Warteschleife gesteckt werden, bevor das eigentliche JavaScript von Google Analytics geladen ist. Sobald das Google-Analytics-Script geladen ist, wird das `_gaq`-Array von diesem überschrieben und durch ein Objekt ersetzt, das ab jetzt angegebene Befehle direkt umsetzen kann.

```
_gaq.push(['_setAccount', 'UA-XXXXX-X']);
```

Dem `_gaq`-Array wird ein Aufruf für die *_setAccount*-Methode übergeben. Dies dient dazu, den Analytics-Account der Website zu bestimmen. Die übergebene UA-Nummer sollte hierbei den Wert Ihres Kontos enthalten.

```
_gaq.push (['_gat._anonymizeIp']);
```

Dem `_gaq`-Array wird mitgeteilt, die Methode *_anonymizeIp* des *_gat*-Objekts aufzurufen, sobald das Google-Analytics-Script geladen ist. Dies ist in Deutschland Pflicht und sorgt dafür, dass die IP nicht vollständig gespeichert wird. Dies ermöglichte Google Analytics unter anderem die Datenschutzkonformität.

```
_gaq.push(['_trackPageview']);
```

Dem `_gaq`-Array wird mitgeteilt, die *_trackPageview*-Methode auszuführen. Dies führt dazu, dass ein Tracking-Pixel an die Google-Analytics-Tracking-Server geschickt wird und später Daten in Ihrer Google-Analytics-Oberfläche erscheinen. Lassen Sie diese Zeile aus Ihrem Snippet heraus, bleibt Ihr Google-Analytics-Konto leer. Es wird also nicht alleine schon deshalb getrackt, weil das JavaScript geladen wird, es muss auch explizit aufgerufen werden!

```
(function() {
  var ga...
})();
```

Der untere Buchstaben-Salat sorgt dafür, dass das eigentliche Google-Analytics-Script geladen wird. Sobald dies fertig geladen ist, werden sämtliche im `_gaq`-Array gespeicherten Methoden aufgerufen und das Array durch ein `_gaq`-Objekt mit voller Funktionalität ersetzt.

## Benutzerdefinierte Variablen

Eine der zusätzlichen Funktionalitäten, die Google Analytics jedem Benutzer zur Verfügung stellt, ist die Möglichkeit, unterschiedliche benutzerdefinierte Variablen festzulegen. Diese werden für jeden einzelnen Besucher gespeichert und können mit beliebigen Werten belegt werden. Da dies eine Menge von Daten erzeugen kann, sind pro Besucher maximal fünf benutzerdefinierte Variablen erlaubt, setzen Sie diese also mit Bedacht ein.

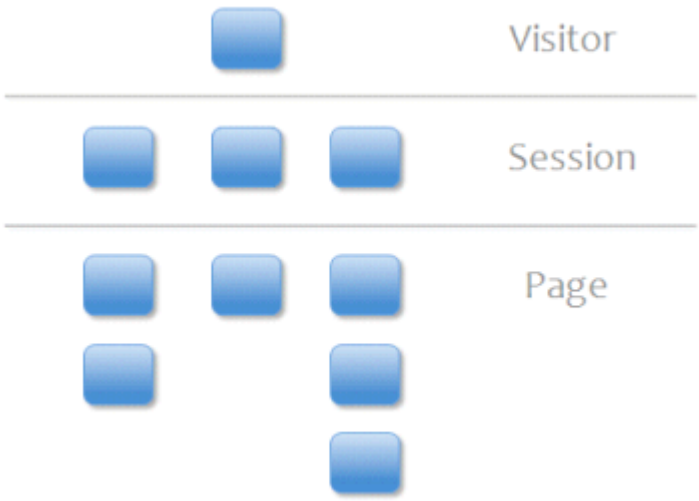



Abbildung 6.11: Aufbau der benutzerdefinierten Variablen (Quelle: https://developers.google.com/analytics/devguides/collection/gajs/gaTrackingCustomVariables)

Ein wichtiger Aspekt der benutzerdefinierten Variablen ist ihr *Scope*. Dieser bestimmt den Geltungsbereich und somit die Reichweite, Wirkungsdauer und Veränderbarkeit der Variablen:

- **Besucher-Ebene:** Ist die benutzerdefinierte Variable auf Besucher-Ebene gesetzt, so bleibt diese für die Lebensdauer des Tracking-Cookies erhalten. Änderungen des Wertes einer solchen Variablen gelten rückwirkend, auch in der Google-Analytics-Oberfläche. Dies kann genutzt werden, um die Anzahl der Einkäufe eines Nutzers mit diesem zu verbinden. Jedes Mal, wenn ein Besucher erneut einkauft, wird dieser Wert um eins erhöht. Da sich die Werte rückwirkend ändern, erhalten Sie auf Google Analytics eine einfache und schnelle Übersicht, wie viele Ihrer Besucher wie oft einkaufen. *Es mag auch durchaus bessere Methoden geben, dies zu erkennen, aber ich hoffe, es zeigt die Funktionsweise dieser Art von Variablen.*
- **Besuchs- oder Session-Ebene:** Ist eine benutzerdefinierte Variable auf Session-Ebene gesetzt, kann ihr Wert nur innerhalb einer Session geändert werden. Jeder neue Besuch ändert alte Werte nicht mehr. Als Session gilt für Google Analytics ein Besuch, bei dem der jeweils letzte Seitenaufruf weniger als 30 Minuten

zurückliegt und sich die Kampagne nicht geändert hat (mehr zu Kampagnen später). Surfen Sie also auf einer Website und machen über 30 Minuten nichts und fangen dann wieder an, sich weiter auf der Seite zu bewegen, sind Sie für Google Analytics ein neuer Besucher. Einsatzzweck für eine benutzerdefinierte Variable auf Session-Ebene kann der Login-Zustand eines Nutzers sein. Auf diese Weise können Sie feststellen, wie viele Ihrer Nutzer als Mitglieder auf Ihrer Seite surfen und wie sich ihr Nutzungsverhalten von nicht eingeloggten Nutzern unterscheidet.

- **Seiten- oder Page-Scope:** Die kurzlebigste Ebene für benutzerdefinierte Variablen ist auf Seiten-Ebene. Dies bedeutet, jeder neue Seitenaufruf führt zu einem neuen Wert, der von Google Analytics für diesen Nutzer gespeichert wird, sofern eine benutzerdefinierte Variable gesetzt ist.

Neben dem Scope können noch weitere Parameter beim Setzen einer benutzerdefinierten Variable übergeben werden:

- Index – Welcher der fünf Speicherplätze für Variablen genutzt werden soll, kann also Werte zwischen 1 und 5 enthalten.
- Name – Der Name für die benutzerdefinierte Variable, dieser erscheint später auch in der Google-Analytics-Oberfläche und sollte stets denselben Wert enthalten.
- Wert – Der Wert, der für diese Variable gespeichert werden soll.

Die Parameter müssen auch in dieser Reihenfolge übergeben werden:

```
_gaq.push(['_setCustomVar', Index, Name, Wert, Scope]);
```

Ein beispielhafter Aufruf zum Setzen einer benutzerdefinierten Variablen sieht so aus:

```
_gaq.push(['_setCustomVar', 5, 'Facebook Besucher-Status', 'FB-Nutzer',
2]);
```

Hier wird die Variable mit Index `5` genutzt und der Name auf `Facebook Besucher-`

`Status` gesetzt, der Wert ist `FB-Nutzer` und als Scope wird durch die `2` die Besuchs-/ Session-Ebene ausgewählt.

> **Wichtig**
>
> Google Analytics schickt nicht bei jedem Methodenaufruf direkt Daten an den Tracking-Server. Das Setzen einer benutzerdefinierten Variablen ist ein solcher Fall. Um sicherzugehen, dass Ihre Daten dennoch übertragen werden, muss die Definition vor einem Aufruf geschehen, der dafür sorgt, dass die Daten übermittelt werden. Methodenaufrufe, die Daten übermitteln, sind _trackPageview, _trackEvent und _trackSocial.

## Eventtracking mit Google Analytics

Eine weitere Funktionalität von Google Analytics ist das Messen von Ereignissen, das so genannte Event-Tracking. Dies kann hilfreich sein, wenn Sie messen wollen, ob Nutzer mit gewissen Teilen auf Ihrer Website interagieren. So kann auf diese Weise gemessen werden, ob ein Besucher ein Video ansieht, eine gewisse Infobox anklickt oder eine Datei herunterlädt.

Dazu muss beim Auftreten dieses Ereignisses die `_trackEvent`-Methode des `_gaq`-Objekts mit den für das Ereignis vorgesehenen Parametern aufgerufen werden:

- Kategorie – Die Kategorie des Events, für ein Video-Tracking ist hier *Video-Tracking* passend.
- Aktion – Mit Aktion wird das Ereignis als solches beschrieben. Im Video-Tracking-Fall könnte ein Video also *gestartet*, *pausiert* oder *gestoppt* werden.
- Bezeichner – Der Bezeichner für das Ereignis, hier würde der Titel des Videos passen, um später identifizieren zu können, mit welchem Video interagiert wurde.
- Wert – Zusätzlich kann noch ein numerischer Wert mitgegeben werden, diese Werte werden jedoch zusammenaddiert und sind nur als Summe später in der

Oberfläche erhältlich. Für ein Video-Tracking kann hier je Nutzer ein Wert zwischen 0 und 100 mitgegeben werden, um anzuzeigen, wie viel Prozent des Videos gesehen wurde.

- Nicht-Interaktiv – Da der Aufruf eines Events als Aktion eines Nutzers betrachtet wird, sorgt dies dafür, dass der entsprechende Nutzer nicht mehr als *Bounce* für die Seite zählt. Da die Bounce-Rate jedoch eine wichtige Metrik ist, kann dies fatal sein. In einigen Fällen können Events genutzt werden, um direkt beim Ladevorgang der Seite Informationen an den Tracking-Server zu schicken. Dies ist jedoch keine Interaktion eines Nutzers und das Verringern der Bounce-Rate aufgrund dieser Tatsache würde diese Metrik unbrauchbar machen, deshalb besteht seit relativ Kurzem die Möglichkeit, hier einen Wert mitzugeben, um Google Analytics mitzuteilen, ob ein Event Auswirkungen auf die Bounce-Rate haben soll oder nicht. Ist der Wert nicht gesetzt, wird dies wie ein *false* interpretiert und der Event verhindert den Bounce; soll der Event also nicht die Bounce-Rate verringern, muss der Parameter mit *true* übergeben werden.

Die Reihenfolge, in der die Parameter übergeben werden, ist wie folgt:

```
_trackEvent(Kategorie, Aktion, Bezeichner, Wert, Nicht-Interaktiv)
```

Ein beispielhafter Aufruf eines Events, wie er auch später in einem Beispiel zu sehen ist:

```
_gaq.push(['_trackEvent', 'F2P2 FB-Ref-Tracking', fb_ref, fb_src, null,
true]);
```

In diesem Fall sieht man anschaulich, dass im Falle von Events die Vorgabe für die Reihenfolge der Parameter – *Kategorie*, *Aktion* und *Bezeichner* – nicht zwangsläufig beachtet werden muss. Zwar wird als erster Wert die *Kategorie* mit `F2P2 FB-Ref-Tracking` angegeben, als *Aktion* und *Bezeichner* aber zwei URL-Parameter übergeben.

## Soziale Interaktionen messen

Eine der neuesten Funktionen, die in Google Analytics zur Verfügung steht und die Wichtigkeit sozialer Netzwerke zeigt, dient speziell zum Messen sozialer Interaktionen.

Die Methode, die Ihnen hierfür im `_gaq`-Objekt zur Verfügung steht, heißt `_trackSocial` und ist wie folgt aufgebaut:

```
_gaq.push(['_trackSocial', Netzwerk, SozialeAktion, Ziel-URL,
Seitenpfad]);
```

Die Parameter, die übergeben werden können, bedeuten:

- `Netzwerk` – Das soziale Netzwerk, mit dem eine Interaktion ausgeführt wurde, dies wäre hier Facebook.
- `SozialeAktion` – Die Interaktion des Nutzers. Je nach sozialem Netzwerk kann dies ein Tweet, ein Like oder Ähnliches sein.
- `Ziel-URL` – Die URL, die von dieser Aktion betroffen ist, also durch einen Tweet, ein Like etc. mit anderen geteilt wird.
- `Seitenpfad` – Der Pfad zur Seite, auf der die Interaktion ausgeführt wurde. Oft wird dies dieselbe Seite sein wie die Ziel-URL, auch wird automatisch der aktuelle Seitenpfad übernommen, sofern kein anderer hier definiert ist.

Der Klick auf einen Like-Button könnte diesen Methoden-Aufruf auslösen:

```
_gaq.push(['_trackSocial', 'Facebook', 'Like', targetURL]);
```

Hier ist `Facebook` das soziale Netzwerk, `Like` die Interaktion und der Wert der Variablen `targetURL` die Ziel-URL. Vorher sollte selbstverständlich der Variablen `targetURL` ein entsprechender Wert zugewiesen werden. Wie dies geht, erfahren

Sie im Abschnitt über das Tracking sozialer Interaktionen.

## Alle Funktionen des _gaq-Objekts

Eine vollständige Dokumentation über die Funktionen des _gaq-Objekts finden Sie unter http://code.google.com/intl/de-DE/apis/analytics/docs/gaJS/gaJSApi.html.

## Kampagnen-Tracking

Um festzustellen, welche Besucher-Quelle oder Kampagne für die Konversion eines Nutzers verantwortlich war, oder um andere Effekte gezielt messen zu können, bietet Google Analytics ein Kampagnen-Tracking an. Dazu werden Ihre Besucher in vier unterschiedliche Gruppen eingeteilt:

- Direkt-Zugriffe – Kommt ein Besucher direkt, also durch Eingabe Ihrer Adresse in die Adressleiste oder über ein Bookmark auf Ihre Seite, wird er als *direct*-Traffic gemessen.
- Organische Zugriffe – Kommt ein Besucher über eine Suchmaschine auf Ihre Seite, wird er automatisch als *organic*-Traffic angesehen.
- CPC-Zugriffe – Kam ein Besucher per Klick auf eine Werbeanzeige auf Ihre Website, wird er als *cpc*-Traffic markiert.
- Referral-Zugriffe – Kommt ein Besucher über eine andere Website auf Ihre Website und diese Website ist nicht als CPC-Traffic oder Suchmaschine erkennbar, wird der Besucher dem *referral*-Traffic zugewiesen.

Da Sie dieser Zuteilung nicht zustimmen müssen oder um diese zu erweitern, können Sie das Kampagnen-Tracking individualisieren. Dazu müssen Sie lediglich einige Parameter an die Link-URL anhängen, die zurück zu Ihrer Website führen. Dies ist selbstverständlich nur dann möglich, wenn Sie auch selbst bestimmen können, wie

dieser Link aussieht. Deshalb werden Kampagnen-Trackings vor allem zur Messung von *Newslettern*, *E-Mailings* und *Werbeanzeigen* genutzt. Auf Facebook können Sie jedoch ebenso sämtliche Links mit entsprechenden Parametern versehen, bevor Sie diese als Statusmeldung veröffentlichen.

Die folgenden fünf Parameter stehen Ihnen zum *Vertaggen* Ihrer Links zur Verfügung:

| **Parameter** | **Beschreibung** | **Beispiel** |
|---|---|---|
| utm_source | Mit diesem Parameter wird die Quelle des Traffics angegeben, der später für diese Kampagne gespeichert werden soll. Dies kann die Domain des Traffics sein oder ein leicht zu identifizierender Name. | Facebook, Example.com |
| utm_medium | Das Medium, über das der Nutzer kam | Email, CPC, Tab-Applikation, Wallpost |
| utm_campaign | Mit diesem Parameter geben Sie Ihrer Kampagne einen Namen. Dieser spiegelt am besten das Ziel dieser Kampagne wider. | Sommerschlussverkauf, Unterhosen-Rabatt-Woche, Wallpost-FB-Aktion-März |
| utm_term | Dieser Parameter ist vor allem für Suchen und CPC gedacht und kann mit einem Suchbegriff belegt werden. | Beliebiger Wert |
| utm_content | Dieser Parameter dient zur Bestimmung, welcher Link angeklickt wurde. Haben Sie in einem Angebot zwei unterschiedliche Links und wollen testen, welcher der beiden öfter genutzt wurde, können Sie den beiden Links unterschiedliche Parameter-Werte geben. Für Facebook könnten Sie hier vermerken, ob der Besucher den Link in den Infos, auf der Wall, in einer Applikation oder bei Bildern Ihrer Fanpage gefunden hatte. | Linker-Button, Grüner-Button, Wallpost, Tab-Applikation, Bilder-Link |

Tabelle 6.1: Die Standard-Parameter für das Kampagnen-Tracking

Der einfachste Weg, Ihre URLs für eine Kampagne anzupassen, ist mit dem vorgefertigten Kampagnen-Tracking-URL-Builder von Google, den Sie unter http://support.google.com/analyti... finden.

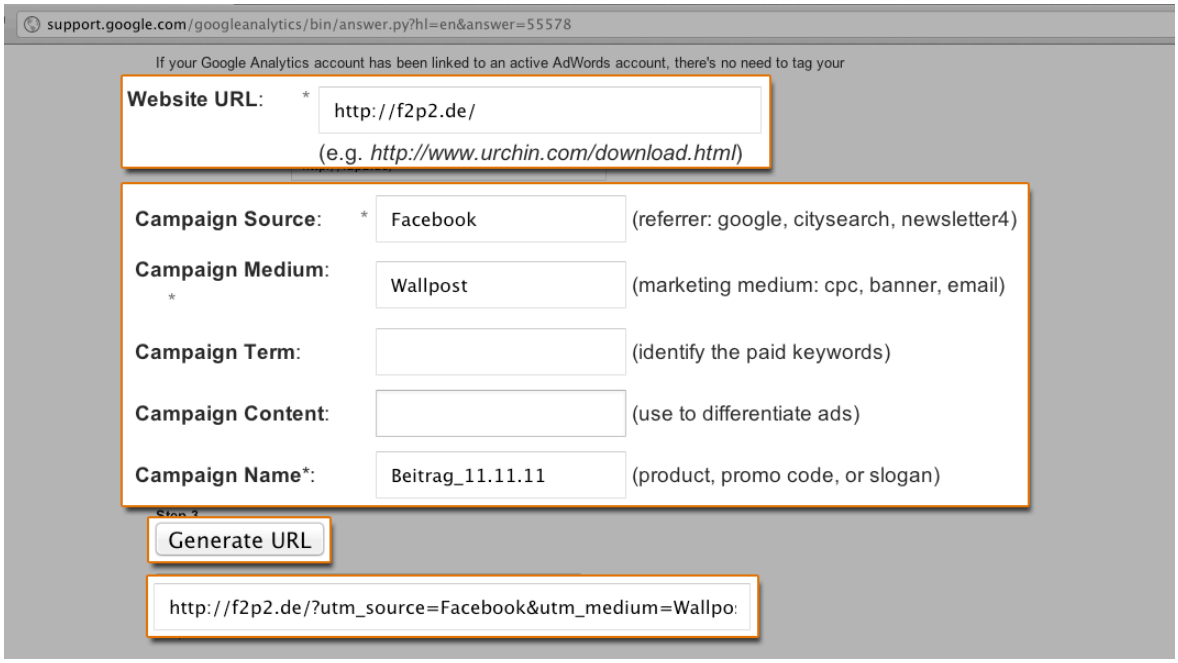



Abbildung 6.12: Campaign-Tracking-Tool von Google

## 2.2. Webreporting-Tools auf Ihrer Facebook-Fanpage

Obwohl der Titel dieses Abschnitts anderes vermuten lässt, ist das Tracken Ihrer gesamten Fanpage nicht möglich.

Durchaus können aber verschiedenste Dinge auf Ihrer Fanpage gemessen werden.

- Da Ihre Tab-Applikationen als Iframe-Elemente eingebunden werden, können diese wie normale Webpages getrackt werden.

- Links, die Sie in Statusmeldungen publizieren, können mit Kampagnen-Tracking versehen werden.
- Links neben geposteten Bildern auf Ihrer Info-Seite oder auf Ihren Applikationen können ebenfalls mit Kampagnen-Tracking versehen werden.
- Nutzer Ihrer Tab-Applikationen können markiert und später auf Ihrer Website wiedererkannt werden: Dies hilft, Erfolg oder Misserfolg einer Kampagne auf Facebook zu erkennen.

## Basis-Tracking der Fanpage-Tabs

Das Tracking Ihrer Fanpage-Tab-Applikationen ist in der Grundform gleich jeder anderen Webpage und kann somit einfach von dort übernommen werden.

Um Ihre Fanpage später besser von normalen Webpages unterscheiden zu können, kann der `_trackPageview`-Methode ein Wert mitgegeben werden, der anstelle des echten Webpagepfads gespeichert wird.

Ist Ihre Tab-Applikation unter `http://example.com/foo/bar/xyz.php?id=123` zu finden, ist dies nur schwer als Tab-Applikation ersichtlich. Geben Sie dem `_trackPageview`-Aufruf den Wert `/facebook-applikation/Impressums-Tab/` mit, wird dieser statt des Originalpfads in der Google-Analytics-Oberfläche angezeigt.

```
<script type="text/javascript">
  var _gaq = _gaq || [];
  _gaq.push(['_setAccount', 'UA-XXXXX-X']);
  _gaq.push(['_trackPageview', '/facebook-applikation/Welcome-Tab/']);

  (function() {
    var ga = document.createElement('script'); ga.type = 'text/
javascript'; ga.async = true;
    ga.src = ('https:' == document.location.protocol ? 'https://ssl' :
'http://www') + '.google-analytics.com/ga.js';
    var s = document.getElementsByTagName('script')[0];
s.parentNode.insertBefore(ga, s);
```

```
    })();
</script>
```

Listing 6.2: Basis-Tracking für eine Fanpage-Tab-Applikation

## Erklärung des Quellcodes

```
_gaq.push(['_trackPageview', '/facebook-applikation/Welcome-Tab/']);
```

Wie bereits erwähnt, ist der einzige Unterschied zu einem herkömmlichen Tracking die `_trackPageview`-Zeile, in der ein Wert übergeben wird, der anstelle des echten Pfads in der Google-Analytics-Oberfläche dargestellt wird.

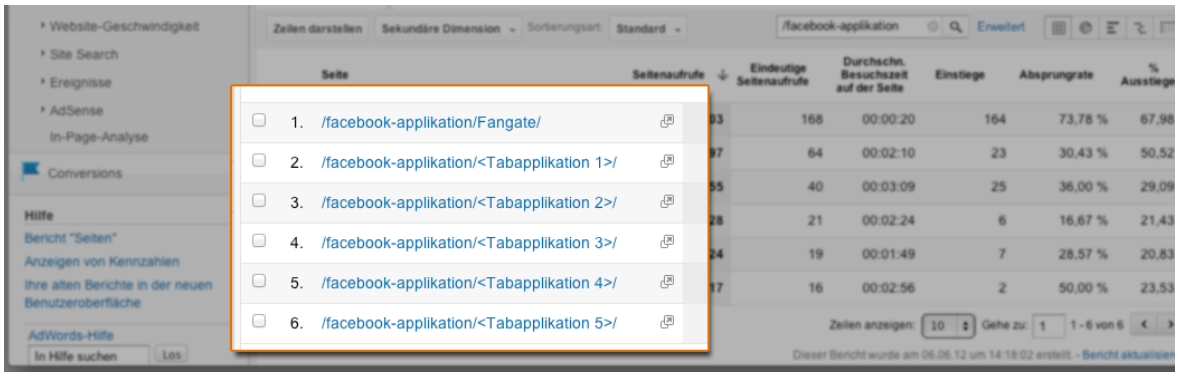



Abbildung 6.13: Übersicht der Besucherzahlen einzelner Tab-Applikationen

## Wen bringt die Fanpage – Wir markieren unsere Herde

Um festzustellen, ob Sie mit Ihrer Fanpage neben altbekannten auch neue Besucher erreichen, sollten Sie messen, wie viele Ihrer Besucher den Erstkontakt mit Ihrer Marke oder Website über Ihre Fanpage hatten.

Dies war zu Zeiten der Welcome-Tabs deutlich einfacher, da garantiert war, dass jeder Besucher, der noch kein Fan Ihrer Fanpage war, zunächst auf Ihre Tab-Applikation geleitet wurde und dort markiert werden konnte. Leider wurden Welcome-Tabs abgeschafft, dennoch sollten Sie diese Messungen weiterhin

vornehmen, gerade dann, wenn Sie Werbekampagnen oder Aktionen auf Facebook haben, die auf einer Tab-Applikation stattfinden.

Wenn Sie diese Nutzer markieren und beobachten, wie viele davon später wieder auf Ihrer Website aufschlagen, können Sie erkennen, ob mit Ihrer Aktion nur Nutzer angelockt wurden, die schnell etwas »abstauben« wollten, oder auch jene, die ernsthaft an Ihrer Marke, Ihrer Website oder den von Ihnen angebotenen Produkten interessiert sind.

Bei der Umsetzung dieser Messung wird zunächst geprüft, ob ein Besucher Ihrem Webreporting-Tool bereits bekannt ist; sollte dies nicht der Fall sein, wird er mit einer benutzerdefinierten Variablen markiert, die ihn später bei einem Besuch auf Ihrer eigentlichen Website wiedererkennbar macht.

Dies bedeutet auch, dass die Trackingcodes auf Ihren Tab-Applikationen und Ihrer Website zusammenspielen müssen. Kommt ein Besucher, den Sie auf Ihrer Tab-Applikation markiert haben, auf Ihre Website, muss der Wert der benutzerdefinierten Variablen angepasst werden, um die Änderung im Status des Besuchers in Google Analytics sichtbar zu machen.

Um das Traffic-Potenzial von Facebook gänzlich messen zu können, wird diese benutzerdefinierte Variable im Verlauf dieses Kapitels noch für weitere Messungen genutzt.

- Kam ein Besucher nach dem ersten Kontakt auf der Fanpage auf die eigene Website
- Kam ein Besucher zum ersten Mal über einen Link aus einer Statusmeldung auf die eigene Website
- Kam ein Besucher über ein Social-Plugin auf die Website

- Kam ein Besucher auf einem nicht messbaren Weg von Facebook auf die Website, also *viral* über Freunde oder geteilte Meldungen

Auf diese Weise erhalten Sie einen klaren Überblick, wie Facebook als Lieferant neuer Besucher dient, und können dies entsprechend einzelnen Kanälen zuweisen. Dies kann genutzt werden, um je nach Zielsetzung Ihre Interaktion mit Ihren Fans zu verbessern. Eine vernünftige Interpretation dieser Daten hilft Ihnen zu erkennen, welche Interaktionen Ihrerseits auf Facebook nachhaltig die meiste Wirkung zeigt, selbst wenn dies nicht immer im ersten Moment sichtbar ist.

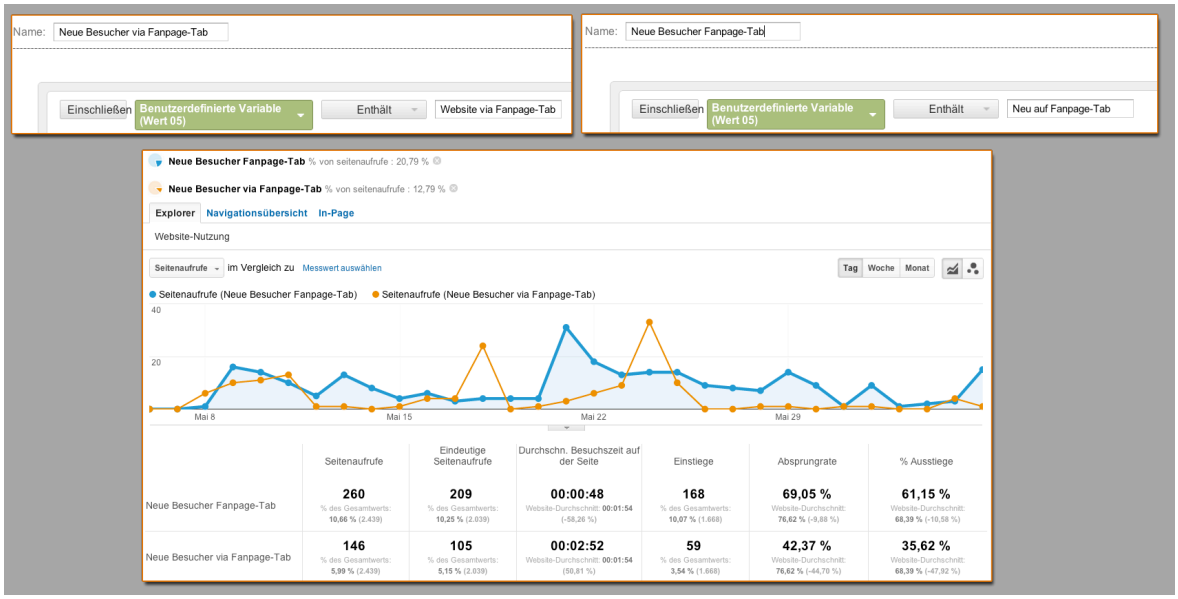



Abbildung 6.14: Übersicht von Fanpage-Tab-Applikations-Besuchern und wie viele davon die eigene Website besuchten

Der Trackingcode zum Markieren neuer Besucher auf Ihrer Fanpage sieht folgendermaßen aus.

```
;(function(){
  window._gaq = window._gaq || [];
  var has_cookie;

  var get_cookie = function(name,all){
```

```
      var cookie_array = document.cookie.split(";")
      ,   cookie_len = cookie_array.length
      ,   index = 0
      ,   name = RegExp("^\\s*" + name + "=\\s*(.*?)\\s*$")
      ,   match_array = []
      ;
      for(;index < cookie_len; index = index + 1){
        var matched_value = cookie_array[index].match(name);
        matched_value && match_array.push(matched_value[1]);
      }
      return all ? match_array : match_array[0];
    };

    has_cookie = !!get_cookie('__utma');

    if(!has_cookie){
      _gaq.push(['_setCustomVar', 5,'Facebook Besucher','Neu auf Fanpage-
Tab',1])
    }
}())
```

Listing 6.3: Google-Analytics-Snippet zur Markierung von Facebook-Fanpage-Besuchern auf den Fanpage-Tab-Applikationen

## Erklärung des Quellcodes

```
window._gaq = window._gaq || [];
```

Das _gaq-Objekt wird initialisiert.

```
var get_cookie = function(name,all){
  ...
}
```

Der Variablen `get_cookie` wird eine Helfer-Funktion zugewiesen, die Cookies auslesen kann und zurückgibt. Dies wird benötigt, um zu prüfen, ob ein Google-Analytics-Cookie bereits existiert und der Nutzer somit bereits bekannt ist.

```
if(!get_cookie('__utma')){
  _gaq.push(['_setCustomVar', 5,'Facebook Besucher ', 'Neu auf Fanpage-
Tab',1])
}
```

Existiert noch kein `__utma`-Cookie und ist der Wert der `has_cookie`-Variablen falsch, wird eine benutzerdefinierte Variable initialisiert. In diesem Beispiel wird der

Index der Variablen auf 5 gesetzt, dieser kann frei gewechselt werden, Sie sollten lediglich darauf achten, bei der Implementierung weiterer Beispiele die Indizes der Variablen ebenfalls anzupassen.

Ist der Trackingcode auf Ihren Tab-Applikationen implementiert, muss der folgende Code auf Ihrer »normalen« Website eingebaut werden, um nach markierten Nutzern Ausschau zu halten.

```
;(function(){
  window._gaq = window._gaq || [];

  _gaq.push(function() {
    var custom_var = _gat._getTrackerByName()._getVisitorCustomVar(5);
    if(!!custom_var && custom_var === 'Neu auf Fanpage-Tab'){
        _gaq.push(['_setCustomVar', 5, 'Facebook Besucher','Website via
Fanpage-Tab',1])
    }
  });
}())
```

Listing 6.4: Google-Analytics-Snippet zur Markierung von Facebook-Fanpage-Besuchern auf der eigenen Website

Die Erklärung zu diesem Code

```
window._gaq = window._gaq || [];
```

Das `_gaq`-Objekt wird initialisiert.

```
_gaq.push(function(){
  ...
});
```

Es wird eine Funktion in das `_gaq`-Objekt gepusht.

```
var custom_var = _gat._getTrackerByName()._getVisitorCustomVar(5);
```

Innerhalb der Funktion wird über den Tracker der Wert für die benutzerdefinierte Variable mit Index 5 abgefragt, also jene Variable, die auf der Fanpage gesetzt wurde, und der Variablen `custom_var` zugewiesen.

```
if(!!custom_var && custom_var === 'Neu auf Fanpage-Tab'){
    _gaq.push(['_setCustomVar', 5, 'Facebook Besucher', 'Website via Fanpage-Tab',1])
}
```

Existiert ein Wert für die benutzerdefinierte Variable und ist dieser gleich dem Wert, den jene Besucher bekommen haben, die als Erstkontakte auf Ihre Tab-Applikation markiert wurden, wird der Wert der benutzerdefinierten Variablen auf einen neuen Wert geändert.

Wie Abbildung 6.14 zeigt, sehen Sie jetzt in der Google-Analytics-Oberfläche die Zahlen für neue Besucher, die nur Ihre Fanpage sahen, und jene, die nach Ihrer Fanpage auch auf Ihrer Website gelandet sind. Vergessen Sie nicht, dass zur Berechnung der Gesamtzahl neuer Besucher beide Werte addiert werden müssen.

## Das Problem IE und die Lösung P3P

Alles wäre schön und gut und Sie könnten genüsslich zuschauen, wie sich Ihr Webreporting-Tool mit Daten füllt, wenn, ja wenn da nicht der Internet Explorer wäre. Wie so oft passt ihm das alles nicht und er mag deshalb nicht so einfach mitspielen. Internet Explorer vor Version 9 verbieten erst einmal alle Third-Party-Cookies und da Ihre Website per Iframe-Element eingebunden ist, ist dies der Fall. Sie können also keine Cookies schreiben und so würden sämtliche Besucher mit einem solchen Browser nicht von Ihrem Tracking erfasst werden.

Glücklicherweise ist das aber nicht das Ende der Geschichte, denn es gibt die guten und ziemlich nutzlosen *P3P-Header*. Diese sollten ursprünglich einmal dazu dienen, dem Browser und somit dem Nutzer mitzuteilen, wofür Cookies auf der Seite genutzt werden und wie dessen Privatsphäre-Einstellungen sind. Eine nette Idee, mehr aber auch nicht.

Um es kurz zu machen, alles, was getan werden muss, damit der Internet Explorer Ihre Cookies akzeptiert, ist, ihm einen P3P-Header zuzuschicken.

Falls Sie sich jetzt fragen, was das Ganze soll und wieso man selbst einen Header schicken kann, um sich selbst das Setzen von Cookies zu erlauben, fragen Sie nicht mich. Aber genau dies dürfte einer der Gründe dafür sein, dass kein Browser außer dem Internet Explorer überhaupt an diesem Header interessiert ist.

Eine »legitime« Implementation eines P3P-Headers kann in PHP wie folgt als Teil des Headers mitgeschickt werden.

```
<?php
header('P3P: CP="CURa ADMa DEVa PSAo PSDo OUR BUS UNI PUR INT DEM STA
PRE COM NAV OTC NOI DSP COR"')
?>
```

Jede dieser Buchstabenkombinationen hat eine Bedeutung, die aber absolut irrelevant ist, da Cookies auch dann akzeptiert werden, wenn dort ein beliebiger Wert eingetragen ist. Der P3P-Header, den Facebook bei seinen Social-Plugins mitschickt, sieht übrigens so aus:

```
P3P: CP="Facebook does not have a P3P policy. Learn why here: http://
fb.me/p3p"
```

Der Inhalt des Headers, der auf die Privatsphäre-Einstellungen der Seite hinweisen soll, besteht also aus einem Hinweis, dass es diese nicht gibt. Oh sweet irony.

```
Expires: Sat, 01 Jan 2000 00:00:00 GMT
P3P: CP="Facebook does not have a P3P policy. Learn why here: http://fb.me/p3p"
Pragma: no-cache
```

Abbildung 6.15: Der P3P-Header von Facebook

## Kampagnen-Tracking für gepostete Links

Um die Reichweite der eigenen Nachrichten zu messen und festzustellen, wie viele Ihrer Fans Ihre Inhalte nicht nur sehen, sondern sich auch dafür interessieren, sollten sämtliche Links in Beiträgen und Statusmeldungen Ihrer Fanpage mit Kampagnen-Parametern versehen werden.
Ein Beispiel für die Kampagnen-Parameter eines Links in einer Statusmeldung wäre wie folgt:

- Die Quelle der Kampagne ist *Facebook*.
- Das Medium der Kampagne ist *Statusmeldung*.
- Dem Kampagnennamen wird ein Wert zugewiesen, der den Post als solchen identifizierbar macht. Dies könnte eine Mischung aus Titel und Datum sein: *post-Titel-xyz_13.03.12*.
- Wenn in einer einzelnen Statusmeldung verschiedene Links vorkommen, können für diese einzelne Content-Tags angelegt werden. Ein Link am Ende einer Statusmeldung hätte dann den Wert *link_unten*.

Kombiniert man diese Parameter miteinander und fügt sie dem ursprünglichen Link hinzu, sieht dies wie folgt aus:

```
http://www.f2p2.de/foo/bar?
utm_source=facebook&utm_medium=statusmeldung&utm_campaign=post-Titel-
xyz_13.03.12&utm_content=Link_unten
```

Neben den Daten, die Ihnen zu jedem Beitrag in den Insights zu Ihrer Fanpage zur Verfügung stehen, erhalten Sie diese jetzt kombiniert mit Ihren sonstigen Webreporting-Daten in der Google-Analytics-Oberfläche.

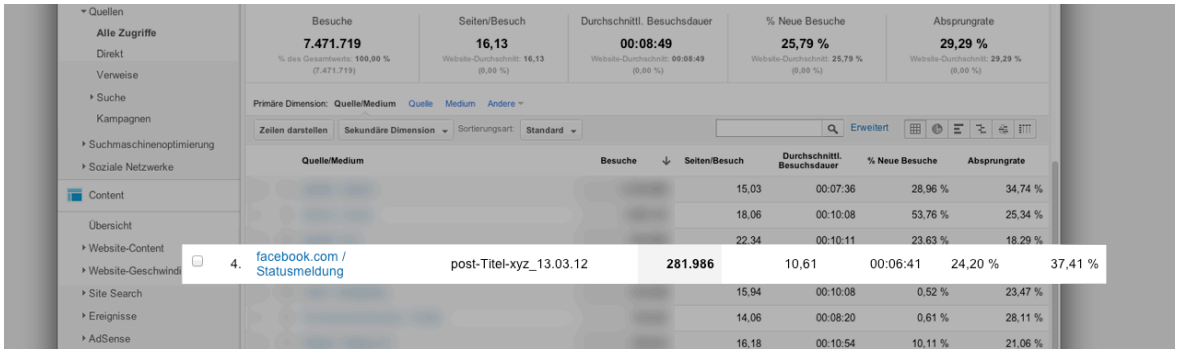


Abbildung 6.16: Darstellung einer getrackten Kampagne über Facebook in der Google-Analytics-Oberfläche

## 2.3. Auf der eigenen Website

Noch weitaus größer als auf der Fanpage sind die Einsatzmöglichkeiten eines Webreporting-Tools auf der eigenen Website. Hier können Sie eine Vielzahl verschiedener Daten erheben, die bei Ihren Besuchern in Verbindung mit Facebook anfallen und messbar werden. In diesem Abschnitt erfahren Sie, wie Sie

- den Facebook-Nutzer-Anteil unter den Besuchern Ihrer Website messen
- soziale Interaktionen Ihrer Nutzer messen
- feststellen, wie viele Nutzer erstmalig über Facebook auf Ihre Website kamen und über welche Quellen
- das `ref`-Attribut Ihrer Social-Plugins tracken
- mit Hilfe des `ref`-Attributs die optimale Position Ihrer Like-Buttons finden

## Ein Neue-Besucher-von-Facebook-Segment einrichten

Auch ohne benutzerdefinierte Variable können Sie herausfinden, wie viele Ihrer Besucher das erste Mal durch Facebook auf Ihre Seite kamen. Dies ist zwar weniger genau als das hier vorgestellte Tracking, ermöglicht Ihnen aber, mit historischen Daten zu arbeiten.
Hierfür wird ein erweitertes Segment in Ihrer Google-Analytics-Oberfläche angelegt.

1. Loggen Sie sich in Ihr Google-Analytics-Konto ein.
2. Klicken Sie auf ERWEITERTE SEGMENTE im oberen linken Bereich der Seite.
3. Wählen Sie im unteren rechten Bereich des jetzt dargestellten Overlays den Knopf +NEUES BENUTZERDEFINIERTES SEGMENT.
4. Geben Sie für dieses Segment einen passenden Namen ein, zum Beispiel NEUE BESUCHER ÜBER FACEBOOK.
5. Wählen Sie als erste Dimension QUELLE aus und geben Sie an, dass diese den Wert FACEBOOK.COM enthalten muss.
6. Ergänzen Sie den Filter mit einer UND-ANWEISUNG und wählen Sie die Dimension BESUCHERTYP, dieser weisen Sie den Wert NEW VISITOR zu.
7. Speichern Sie Ihre Angaben durch einen Klick auf SEGMENT SPEICHERN.

Google Analytics erstellt jetzt ein Segment an Besuchern, die diesen Kriterien entsprechen.

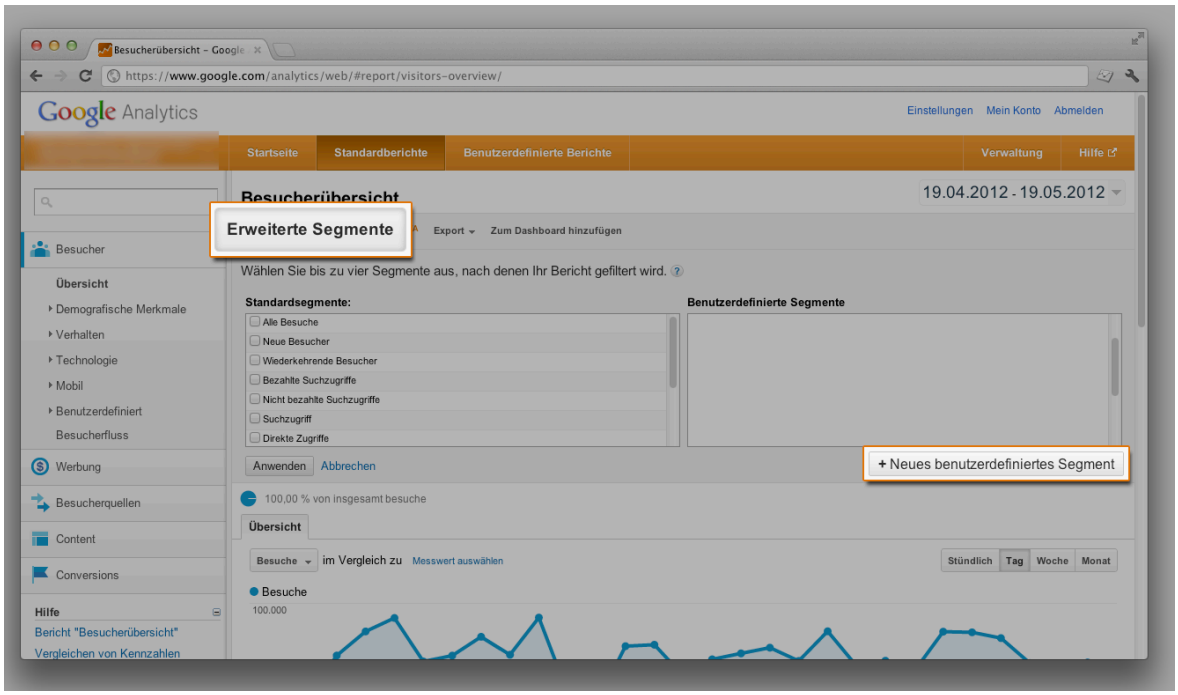



Abbildung 6.17: Step 1: Erweiterte Segmente anzeigen und ein neues erstellen

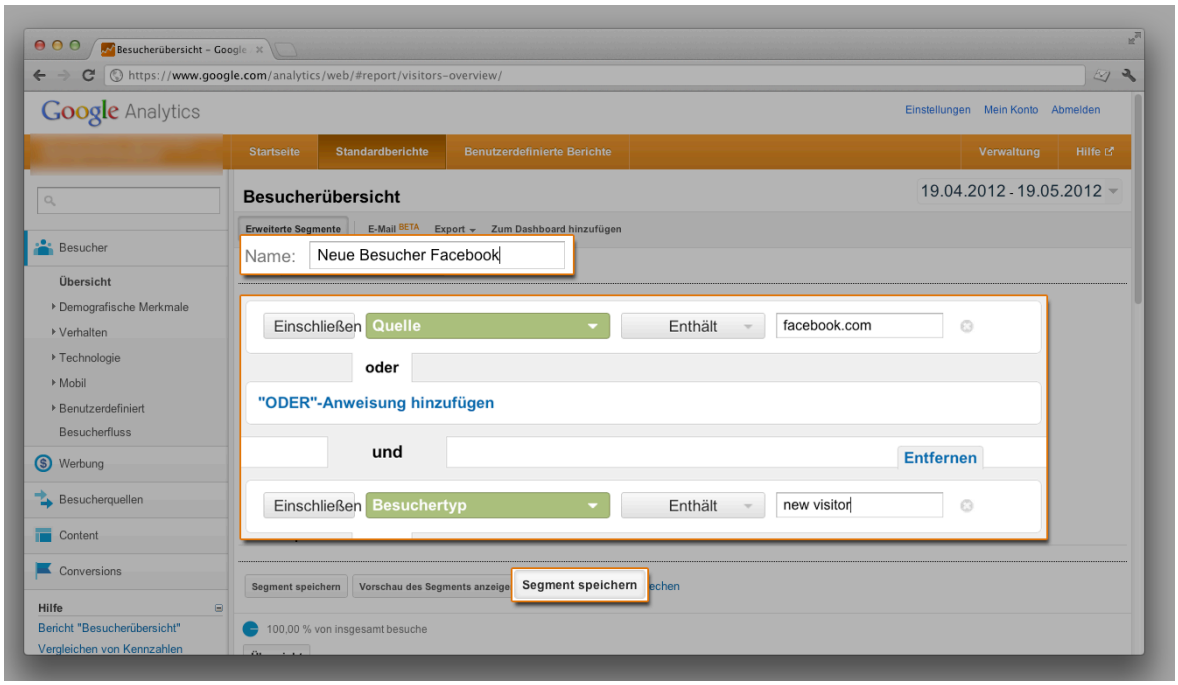



Abbildung 6.18: Step 2: Filter für das Segment einrichten

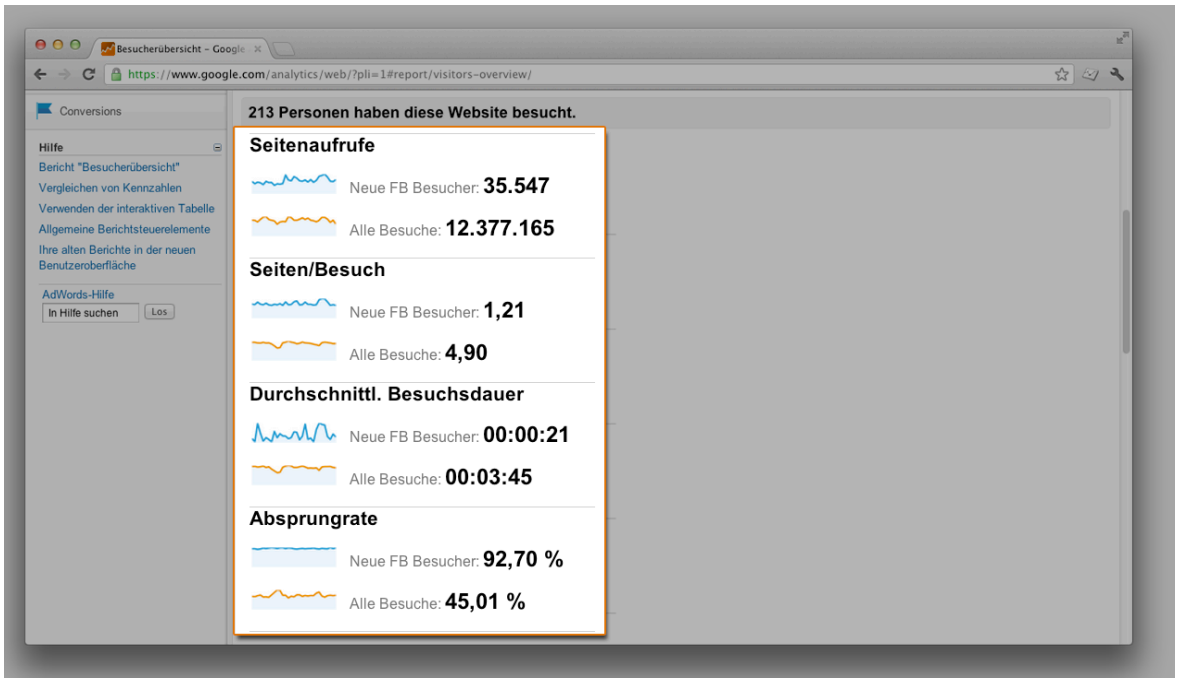



Abbildung 6.19: Step 3: Überrascht sein, wie stark sich die Zahlen unterscheiden

Um Ihre so gewählte Gruppe jetzt mit anderen Besuchergruppen zu vergleichen, klicken Sie einfach erneut auf ERWEITERTE SEGMENTE und wählen Sie ein weiteres Segment aus. Google Analytics bietet Ihnen auch eine Reihe vorgefertigter STANDARDSEGMENTE an. So könnten Sie das Segment ALLE BESUCHER wählen, um Ihre Facebook-Erstkontakte gegen alle anderen Besucher zu vergleichen.

## Tracking des Ref-Attributs

Wie in Kapitel 5 *(in diesem Ausschnitt nicht enthalten)* beschrieben, steht einigen Social-Plugins das `ref`-Attribut zur Verfügung. Wird ein Social-Plugin mit diesem Attribut und das Attribut mit einem Wert versehen, so wird dieser Wert an alle Links, die zurück zu Ihrer Website führen, als `fb_ref`-Parameter an die URL angehängt. Zusätzlich und abhängig vom Social-Plugin wird von Facebook ebenfalls ein `fb_source`-Parameter hinzugefügt, dieser enthält Informationen darüber, wo der

Link auf Facebook gefunden wurde.

Durch das Tracking des `ref`-Attributs erhalten Sie Zugang zu einer Reihe interessanter Informationen:

Sie erhalten die Möglichkeit, zu erkennen,

- wie viele Besucher von Ihren Social-Plugins generiert werden
- im Detail, welche Social-Plugins den meisten Traffic generieren
- welche Seiten und somit welche Inhalte für Besucher am attraktivsten waren und angeklickt wurden
- aufgrund der Absprungrate und Verweildauer dieser Besucher, ob der Teaser, den die Besucher zu Ihrer Website gesehen haben, angemessen war und die Erwartungen der Besucher erfüllt wurden oder die Darstellung zu plakativ war und Ihre Besucher gleich wieder verschwunden sind
- die Möglichkeit, die Position und das Aussehen Ihrer Social-Plugins zu testen und so zu optimieren, dass Ihren Besuchern das Teilen leichter fällt

Mit diesen Informationen ausgestattet können Sie die Gestaltung Ihrer Teaser so optimieren, dass jeder Like, der von Ihrer Website auf Facebook erscheint, eine maximale Anzahl an Besuchern zurück auf Ihre Website lockt und dort zu glücklichen Besuchern werden lässt.

## Das ref-Attribut vorbereiten

Da der `fb_ref`-Parameter von Haus aus keine Rückschlüsse auf das Social-Plugin zulässt, das ihn erstellt hat, sollten Sie dies im Wert des Attributs vermerken.
Wenn Sie zusätzlich noch weitere Informationen über das `ref`-Attribut transportieren wollen, sollten Sie dafür sorgen, die einzelnen Daten klar voneinander zu trennen, um sie getrennt voneinander später wieder auswerten zu können.
Gehen wir davon aus, Sie haben einen Like-Button auf Ihrer Website. Dieser soll

sowohl Informationen über das Plugin selbst, dessen Position und einen weiteren Wert enthalten. Damit diese Daten später wieder einfach voneinander getrennt und verarbeitet werden können, müssen diese mit einem Separator voneinander getrennt werden, der später wieder erkannt werden kann. Als Beispiel wird hierfür ein doppeltes Pipe-Symbol genutzt.

Hat das `ref`-Attribut also den Wert `Like-Button||Links-Oben||XYZ` und ein Besucher klickt darauf, wird dem Link zurück auf die Website dieser Wert als `fb_ref`-Parameter angehängt. Ein Link auf `http://www.f2p2.de`, der im News-Stream eines Facebook-Nutzers erscheint, führt somit auf folgende URL:

http://www.f2p2.de/?fb_ref=Like-Button||Links-Oben||XYZ&fb_source=home

Hier kann der Wert des `fb_ref`-Parameters aus der URL ausgelesen und anhand des Separators leicht in seine Einzelteile zerlegt und weiterverarbeitet werden.

## fb_ref- und fb_source-Tracking

Da das Tracking der Parameter leider nicht von alleine stattfindet, muss auf Ihrer Website ein entsprechender Trackingcode angelegt werden. Dieser Code soll prüfen, ob in der aktuellen URL ein entsprechender Parameter existiert; ist dies der Fall, muss ein Event ausgelöst werden, der Google Analytics die vorher formatierten Werte des Parameters übermittelt.

```
;(function(){
  /*
  * Helper function to extract a specific query parameter from a URL.
  * Source: http://code.google.com/apis/analytics/docs/tracking/
gaTrackingSocial.html#twitter
  *
  * @param  uri        The URL to extract the parameter from
  * @param  parameter  The name of the parameter to look for
  * @return null or value of the parameter
  */
```

```
  var extractParameter = function (uri, parameter) {
    if (!uri) {
      return;
    }

    var uri = uri.split('#')[0]
    ,    parts = uri.split('?')
    ,    i = 0
    ,    query
    ,    params
    ,    param
    ;

    if(parts.length === 1){
      return;
    }

    query = decodeURI(parts[1]).replace(';', '&');
    parameter += '=';
    params = query.split('&');

    for(; param = params[i]; i += 1) {
      if (param.indexOf(parameter) === 0) {
        return unescape(param.split('=')[1]);
      }
    }

    return;
  };

  var fb_src = extractParameter(window.location.href, "fb_source") ||
"not set"
  ,    fb_ref = extractParameter(window.location.href, "fb_ref")
  ;

  if(!fb_ref){
    return;
  }
  window._gaq = window._gaq || [];
  window._gaq.push(['_trackEvent', 'F2P2 FB-Ref-Tracking', fb_ref,
fb_src, null,
true]);
}())
```

Listing 6.5: Tracking-Funktion für das fb_ref- und fb_source-Attribut

Erklärung des Codes:

```
var extractParameter = function (uri, parameter) {
  ...
};
```

Eine Helper-Funktion, die URL-Parameter extrahieren kann.

```
var fb_src = extractParameter(window.location.href, "fb_source") ||
"not set"
```

Die Variable `fb_src` wird mit dem Wert des `fb_source`-Parameters belegt. Existiert dieser Parameter nicht, wird der Variablen stattdessen der Wert *not set* zugewiesen.

```
,    fb_ref = extractParameter(window.location.href, "fb_ref")
```

Der Variablen `fb_ref` wird der extrahierte Wert des `fb_ref`-Parameters zugewiesen.

```
if(!fb_ref){
   ...
}
```

Ist die Variable `fb_ref` leer und somit ohne Wert, wird der Tracking-Vorgang abgebrochen.

```
window._gaq = window._gaq || [];
window._gaq.push(['_trackEvent', 'F2P2 FB-Ref-Tracking', fb_ref,
fb_src, null, true]);
```

Existiert der `fb_ref`-Parameter, wird ein Event an das `_gaq`-Objekt übergeben. Dieser enthält sowohl den Wert des `fb_ref`-Parameters als auch des `fb_src`-Parameters. Der letzte übergebene Parameter *true* sorgt dafür, dass Google Analytics diesen Event als nicht-interaktiv ansieht und somit Ihre Bounce-Rate nicht zerstört.

## Tracking sozialer Interaktionen

Neben statischen, also bereits beim Ladeprozess der Seite bekannten, Metriken können ebenso eine Reihe dynamischer Daten erfasst werden, die erst während eines Besuches entstehen. Dies ist in Verbindung mit Facebook dann der Fall, wenn ein Nutzer mit einem Ihrer Social-Plugins interagiert.

Das Messen dieser Interaktionen ist jedoch nicht ohne Weiteres möglich. Da sich Ihre Social-Plugins in Iframe-Elementen befinden und Sie keinen Zugriff auf diese haben, können Sie auch nicht feststellen, wann ein Nutzer dort geklickt oder eine Aktion ausgeführt hat. Aus diesem Grund bietet Facebooks JavaScript-SDK eine Möglichkeit, über Interaktionen Ihrer Besucher mit Social-Plugins informiert zu werden. Dazu muss eine so genannte Callback-Funktion an das JavaScript-SDK übergeben und für einen Event registriert werden. Tritt ein Event, also eine Interaktion, auf, wird diese Callback-Funktion aufgerufen.

Die Callback-Funktion selbst muss jetzt nur noch erkennen, wer für Ihren Aufruf verantwortlich war, und die entsprechenden Daten an Google Analytics weiterleiten.

Die Methode, mit der Sie Events abonnieren können, heißt `FB.Event.subscribe` und erlaubt das Abonnieren der folgenden Social-Plugin-bezogenen Events:

- *edge.create* – Dieser Event wird ausgelöst, wenn ein Besucher Ihrer Website auf einen Like-Button klickt.
- *edge.remove* – Dieser Event wird ausgelöst, wenn ein Besucher seinen Like wieder rückgängig macht.
- *comment.create* – Dieser Event wird ausgelöst, wenn ein Kommentar auf Ihrer Seite abgegeben wird, dies kann selbstverständlich nur dann passieren, wenn Sie auch ein Comments-Plugin auf Ihrer Seite haben.

- *comment.remove* – Dieser Event wird ausgelöst, wenn ein Besucher seinen abgegebenen Kommentar wieder löscht.
- *message.send* – Dieser Event wird ausgelöst, wenn der Send-Button von einem Besucher genutzt wurde.

## Einfaches Event-Tracking

Zum Messen dieser Interaktionen können Sie auf die speziell von Google Analytics dafür vorgesehene `_trackSocial`-Methode zurückgreifen. Diese erlaubt Ihnen, diverse Aktionen auf Ihrer Website einem sozialen Netzwerk zuzuweisen und gezielt in der Oberfläche wieder einzusehen.

Eine beispielhafte Implementation dieses sozialen Trackings kann für Facebook wie folgt aussehen:

```
;(function(){
  var interactions = {
     'like':            'edge.create'
  ,  'unlike':          'edge.remove'
  ,  'share':           'messages.send'
  ,  'comment':         'comment.create'
  ,  'delete_comment':  'comment.remove'
  }
  ,   interaction
  ;
  for (interaction in interactions) {
    if (interactions.hasOwnProperty(interaction)) {
      FB.Event.subscribe(interactions[interaction],
(function(interaction) {
        return function(targetUrl){
          _gaq.push(['_trackSocial', 'facebook', interaction,
targetUrl]);
        };
      })( interaction));
    }
  }
}())
```

Listing 6.6: Event-Tracking für Interaktionen mit Social-Plugins

Erklärung des obigen Quellcodes:

```
var interactions = {
```

```
        'like':             'edge.create'
      , 'unlike':           'edge.remove'
      , 'share':            'messages.send'
      , 'comment':          'comment.create'
      , 'delete_comment':   'comment.remove'
      }
```

Ein Objekt, das genutzt wird, um später alle Events zu abonnieren, die getrackt werden sollen.

```
      , interaction
```

Eine Helfervariable, die zum Iterieren über das Objekt benötigt wird.

```
for (interaction in interactions) {
  if (interactions.hasOwnProperty(interaction)) {
    ...
  }
}
```

Eine `for`-Schleife, die über jedes *Key-Value-Paar* im Objekt iteriert. Im gleichen Zug wird geprüft, ob das gefundene Attribut auch dem Objekt gehört oder nur dessen Prototypen. Dieses Problem kann hier zwar nur auftreten, wenn der *Prototyp* des globalen Object-Objekts verändert wird, und obwohl dies nicht gemacht werden sollte, kann es aber passieren.

```
FB.Event.subscribe(interactions[interaction], (function(interaction) {
  ...
})( interaction));
```

Die einzelnen Events werden abonniert. Dazu wird der Value-Wert des momentan in der `for`-Schleife gefundenen Attributs als Event-Name angegeben, zum Beispiel *edge.create*, und das Attribut selbst, zum Beispiel *Like*, in einem *Closure* »gefangen«, um das Loop-Problem zu umgehen.

```
    return function(targetUrl){
      _gaq.push(['_trackSocial', 'facebook', interaction, targetUrl]);
    };
```

Wir geben eine Funktion zurück, die jetzt in den Aufruf der FB-Funktion übergeben wird und beim Auftreten des Events dafür sorgt, dass Google Analytics einen sozialen Event trackt.

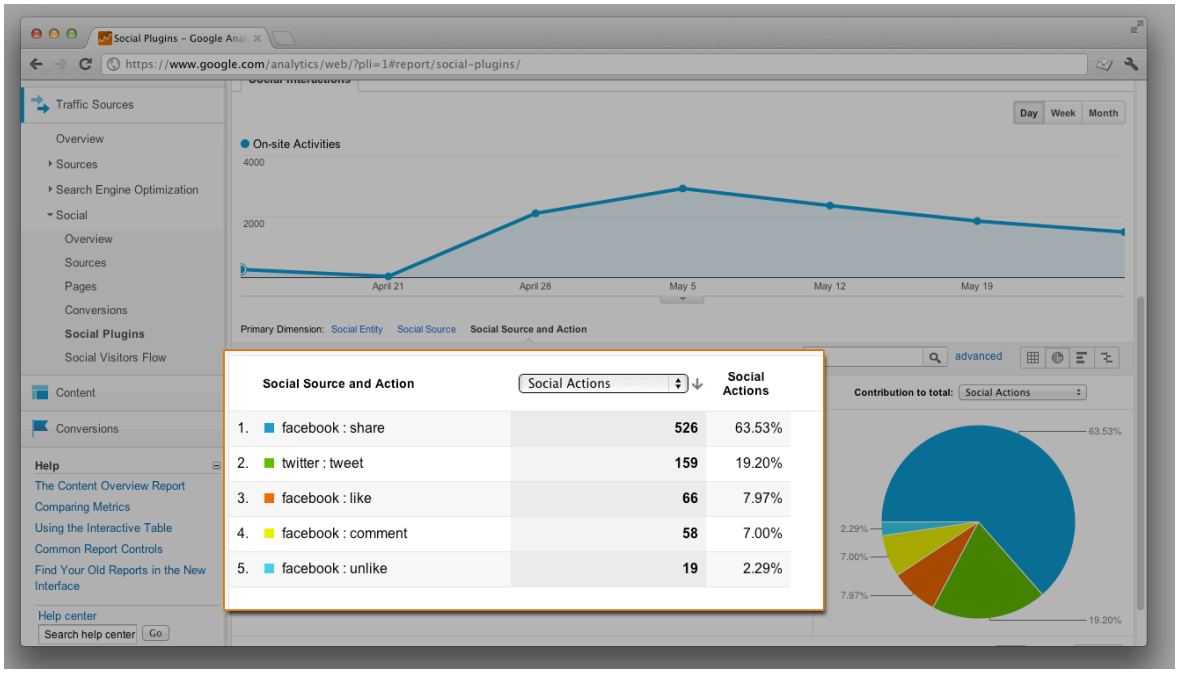



Abbildung 6.20: Anzeige sozialer Interaktionen auf der Website in der Google-Analytics-Oberfläche

## Event-Tracking mit Conversion-Optimierung für Like-Buttons

Hier nun die Begründung meiner in Kapitel 5 aufgestellten Behauptung, dass das `ref`-Attribut nicht wie von Facebook beschrieben zur Optimierung genutzt werden kann und wie es aber dennoch, wenn auch auf anderem Wege geht.

Wenn Sie feststellen wollen, welche Position oder welche Optionen für Ihren Like-Button am geeignetsten sind, muss dies getestet werden. Dazu müssen Sie feststellen, mit welchem Like-Button am häufigsten interagiert wird. Wie Sie bereits

erfahren haben, werden Sie über diese Interaktionen durch Abonnieren der entsprechenden Events informiert, leider aber nicht darüber, von welchem Button dies ausgelöst wurde.

Da Sie die URL erhalten, die im `href`-Attribut des Like-Buttons steht, wäre eine Idee einfach, diese je nach Like-Button ein wenig zu verändern. So könnte man den URLs unterschiedliche Hash-Werte, also Werte nach dem #-Zeichen, verpassen, die eigentlich ignoriert würden. Nicht so aber bei Facebook und so würden Ihre Likes je nach Like-Button unterschiedlichen Webpages zugewiesen werden, ein unvorteilhaftes Verhalten.

Glücklicherweise gibt es auch hierfür einen kleinen Trick. Wenn man zu viel Zeit und Langeweile hat, findet man heraus, dass dem Callback, der aufgerufen wird, wenn ein abonnierter Event auftritt, nicht nur die URL übergeben wird wie dokumentiert, sondern auch ein JavaScript-Objekt. Jedes Social-Plugin wird vom JavaScript-SDK von Facebook intern in einem Objekt angelegt. Das hier übergebene Objekt ist genau ein solches und zwar das des Like-Buttons, der angeklickt wurde, und dieses Objekt wiederum enthält auch den Wert des `ref`-Attributs. Wurde in diesem also die Position des Like-Buttons definiert, kann dies ausgelesen und verarbeitet werden.

Eine Erweiterung des bereits vorgestellten Social-Tracking-Snippets würde entsprechend wie folgt aussehen:

```
;(function(){
  var track_social_events = function(){
    var interactions = {
       'like':            'edge.create'
    ,  'unlike':          'edge.remove'
    ,  'share':           'messages.send'
    ,  'comment':         'comment.create'
    ,  'delete_comment':  'comment.remove'
    }
    ,   interaction
    ;

    for (interaction in interactions) {
      if (interactions.hasOwnProperty(interaction)) {
```

```
        FB.Event.subscribe(interactions[interaction],
(function(interaction) {
          return function(targetUrl){
            if(obj && obj['_attr'] && obj['_attr'].ref){
              _gaq.push(['_trackEvent', 'FB Convtrack', interaction,
obj['_attr'].ref]);
            }
            _gaq.push(['_trackSocial', 'facebook', interaction,
targetUrl]);
          };
        })( interaction));
      }
    }
  };
}())
```

Listing 6.7: Interaktionsmessung von Social-Plugins mit zusätzlicher Conversion-Optimierungs-Messung

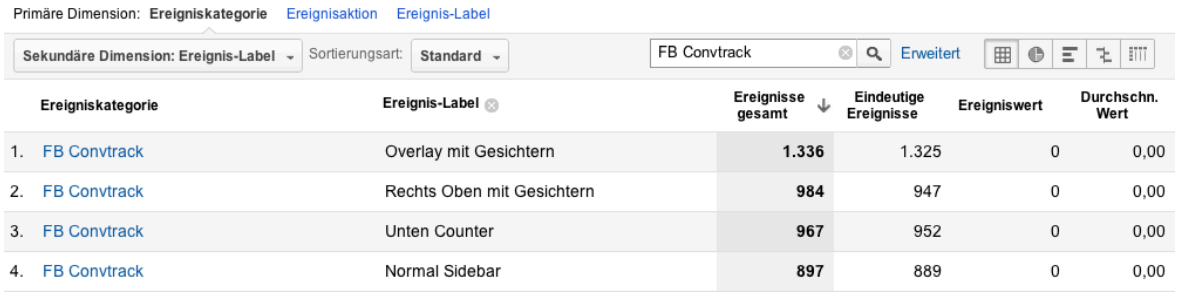
Zusätzlich zum bereits bekannten Interaktions-Tracking kamen die folgenden Code-Zeilen hinzu:

```
if(obj && obj['_attr'] && obj['_attr'].ref){
  ...
}
```

Es wird geprüft, ob das Objekt, das übergeben wurde, ein `ref`-Attribut besitzt. Ist dies der Fall, werden die Befehle innerhalb des Blocks ausgeführt.

```
_gaq.push(['_trackEvent', 'FB Convtrack', interaction,
obj['_attr'].ref]);
```

Ist das `ref`-Attribut vorhanden, wird ein Event an Google Analytics übermittelt. Dieser enthält einen Kategorie-Namen, hier *FB Convtrack*, einen Interaktionsnamen, dies könnte *Like* oder *Send* sein, sowie den Namen des `ref`-Attributs als Label.
In Ihrer Google-Analytics-Oberfläche können Sie jetzt leicht prüfen, welches Ihrer Social-Plugins am besten abgeschnitten hat.



Primäre Dimension: **Ereigniskategorie** Ereignisaktion Ereignis-Label

Sekundäre Dimension: Ereignis-Label | Sortierungsart: Standard | FB Convtrack | Erweitert

| | Ereigniskategorie | Ereignis-Label | Ereignisse gesamt | Eindeutige Ereignisse | Ereigniswert | Durchschn. Wert |
|---|---|---|---|---|---|---|
| 1. | FB Convtrack | Overlay mit Gesichtern | **1.336** | 1.325 | 0 | 0,00 |
| 2. | FB Convtrack | Rechts Oben mit Gesichtern | **984** | 947 | 0 | 0,00 |
| 3. | FB Convtrack | Unten Counter | **967** | 952 | 0 | 0,00 |
| 4. | FB Convtrack | Normal Sidebar | **897** | 889 | 0 | 0,00 |

Abbildung 6.21: Unterschiedliche Erfolge je nach Like-Button-Implementierung

## Den Facebook-User-Anteil auf Ihrer Website messen

Eine der wohl wichtigsten und interessantesten Kennzahlen bei der Webanalyse in Verbindung mit Facebook ist der Anteil aktiver Facebook-Nutzer an den eigenen Besucherzahlen. Eine solche Metrik gibt Ihnen die Möglichkeit, eine Reihe von Rückschlüssen ziehen zu können:

- Wie verhalten sich Facebook-Nutzer im Vergleich zu anderen Nutzern? Gibt es erkennbare Unterschiede bei *Seiten pro Besuch*, *Verweildauer* oder *Bounce-Rate*? Konvertieren Facebook-Nutzer anders, vielleicht sogar besser?
- Erreichen Sie die erhoffte Zielgruppe? Ist das Ergebnis des derzeitigen Facebook-Engagements gut oder sollte die Strategie überdacht werden?
- Verändern eigene Aktionen auf Facebook und der Fanpage merkbar die Daten der eigenen Website oder spielt das eigene Engagement keine große Rolle?

## Woher nehmen, wenn nicht stehlen?

Bleibt also nur noch die Frage, wie Sie an eins diese Daten kommen können. Zwar erhalten Sie in Facebook Insights Informationen über die Menge an Facebook-Nutzern auf Ihrer Website, aber diese Daten lassen sich nicht in Ihr Webreporting-Tool integrieren.

Relativ sicher ist wiederum, dass ein Nutzer auf Facebook ist, sofern dieser über Facebook auf Ihre Website kommt, Sie könnten also sämtliche Besucher, die mit Facebook als Referrer auf Ihre Website kommen, entsprechend über eine benutzerdefinierte Variable auf Besucher-Ebene markieren. Dies bringt aber das Problem, dass sämtliche Besucher, die nicht über Facebook auf Ihre Website kommen, aber dennoch dort eingeloggt sind, nicht erfasst werden können. Auch ist nicht sichergestellt, dass der Nutzer, der einmal über Facebook auf Ihre Seite kam, auch immer dort eingeloggt ist. Um sich ein genaues Bild zu verschaffen, hilft wie so oft also nur ein kleiner Trick.

Sieht man sich das JavaScript-SDK von Facebook genauer an, findet man eine Methode mit dem Namen *getLoginStatus*. Diese gibt Websitebetreibern die Möglichkeit, bei Facebook anzufragen, ob der Besucher, der sich gerade auf der Website befindet, Nutzer einer bestimmten Applikation ist.

Die Applikation, deren Nutzung bei dieser Abfrage geprüft wird, muss beim Initialisieren des JavaScript-SDKs angegeben werden und erklärt auch, wieso diese Funktion existiert.

Facebook-Applikationen können von Websitebetreibern genutzt werden, um Besuchern das Einloggen oder Registrieren über Facebook zu ermöglichen. Da die Website selber aber auch prüfen können muss, ob ein Besucher eingeloggt ist oder nicht, existiert diese Funktion.

Bleibt noch die Frage offen, wie das beim Erreichen des Ziels weiterhelfen soll. Um dies zu klären, müssen lediglich die Antworten dieser Funktion betrachtet werden. Wenn man die `getLoginStatus`-Methode nutzen will, muss man dieser eine Callback-Funktion übergeben, diese wird aufgerufen, sobald die Abfrage an Facebook abgeschlossen ist und der Status übermittelt wurde. Exakt diese

Statusinformationen werden jetzt der Callback-Funktion in einem Response-Objekt übergeben, das die folgenden drei Informationen beinhalten kann:

- `connected` – Facebook hat den Besucher als Nutzer der Applikation erkannt.
- `not_authorized` – Facebook hat den Besucher erkannt. Dieser ist jedoch kein Nutzer der Applikation.
- `unknown` – Facebook kennt den Besucher nicht.

Und genau danach suchen wir, denn diese Statusmeldungen bedeuten nichts anderes, als dass in den ersten beiden Fällen der Besucher von Facebook erkannt wurde, er also eingeloggt ist, im dritten Fall aber nicht.

Alles, was Sie jetzt noch tun müssen, ist, diese Informationen in Form einer benutzerdefinierten Variablen zu speichern und an Google Analytics zu schicken, um jetzt die Facebook-Besucher von Ihren restlichen Besuchern unterscheiden zu können.

## Speichern der Facebook-Anteile auf Session-Ebene

Sicherlich könnte für das Messen des Facebook-Anteils dieselbe benutzerdefinierte Variable genutzt werden, die bereits zum Messen der Erstkontakte über die Fanpage genutzt wurde, da diese Besucher logischerweise ebenfalls Facebook-Besucher sind. Dies birgt allerdings das Problem, dass Besucher, die zum Zeitpunkt des Besuchs gar nicht auf Facebook eingeloggt waren, trotzdem als solche Facebook-Besucher betrachtet werden.
Hier kommt der Vorteil einer Betrachtung auf Session-Ebene zum Zuge, da ausschließlich Besucher als Facebook-Nutzer identifiziert werden, die aktiv zum Zeitpunkt des Besuchs auf Facebook eingeloggt sind.

Die Implementierung eines Trackings auf Session-Ebene sieht wie folgt aus:

```
;(function(){
  var _gaq = _gaq || [];
  FB.getLoginStatus(function(response){
    if (response.status === 'connected' || response.status ===
'not_authorized') {
      _gaq.push(['_setCustomVar', 4, 'Facebook Besucher-Status', 'FB-
Nutzer', 2]);
    }
  });
}())
```

Listing 6.8: Facebook-Nutzer-Anteile unter den Besuchern messen

Genaue Erklärung des Codes:

```
var _gaq = _gaq || [];
```

Das `_gaq`-Objekt wird initialisiert.

```
FB.getLoginStatus(function(response){
  ...
});
```

Die Methode zur Prüfung des Login-Status wird aufgerufen und dieser wird eine Funktion übergeben, die einen *response*-Parameter als Antwort erwartet.

```
if (response.status === 'connected' || response.status ===
'not_authorized') {
  _gaq.push(['_setCustomVar', 4, 'Facebook Besucher-Status', 'FB-
Nutzer', 2]);
}
```

Ist der Status der Antwort `connected` oder `not_authorized`, wird die benutzerdefinierte Variable auf den Wert `FB-Nutzer` gesetzt.

In Ihrer Google-Analytics-Oberfläche erhalten Sie jetzt unter BESUCHER -> DEMOGRAFISCHE MERKMALE -> BENUTZERDEFINIERTE VARIABLEN eine Übersicht über die Login-Zahlen Ihrer Besucher.

## Facebook-Nutzer-Segment anlegen

Neben der reinen Quantität können Sie mit dieser Variablen aber auch das Verhalten dieser Gruppe ins Verhältnis zu Ihren übrigen Besuchern stellen. Loggen Sie sich dazu in Ihr Google-Analytics-Konto ein und wählen Sie im Standard-Reporting ERWEITERTE SEGMENTE.

- Legen Sie mit einem Klick auf + NEUES BENUTZERDEFINIERTES SEGMENT ein neues Segment an.
- Wählen Sie als Dimension Ihres Segments BENUTZERDEFINIERTE VARIABLE (WERT 04), wählen Sie als Vergleich ENTHÄLT und geben Sie den Wert FB an.
- Speichern Sie das Segment mit einem Klick auf SEGMENT SPEICHERN.

Google Analytics zeigt Ihnen jetzt nur noch Daten für Besucher, die in diesen Segment-Filter gelaufen sind.

Wenn Sie diese Daten den allgemeinen Daten gegenüberstellen möchten, klicken Sie erneut auf ERWEITERTE SEGMENTE und wählen Sie zusätzlich das vordefinierte Segment ALLE BESUCHE aus. Sie erhalten eine Gegenüberstellung der beiden Segmente.

## Integration in die Fanpage-Variable

Um neben der Betrachtung auf Session-Ebene zusätzlich eine übergreifende Betrachtung zu erhalten, kann der Login-Status ebenfalls in die bereits vorhandene benutzerdefinierte Variable für Ihre Fanpage integriert werden.
Alles, was der Tracking-Code dazu erkennen muss, ist, ob die Variable bereits einen Wert hat, da dieser nicht geändert werden sollte. Ist dies nicht der Fall, kann der Besucher als Facebook-Nutzer markiert werden.

Das bereits vom Fanpage-Tracking bekannte Script für Ihre Website muss wie folgt erweitert werden:

```
;(function(){
  var _gaq = _gaq || [];
  _gaq.push(function() {
    var custom_var = _gat._getTrackerByName()._getVisitorCustomVar(5)
    ,   ref_is_fb = !!document.referrer.match(/(?:.*)?\.facebook
\.com(?:\/|$)/)
    ,   first_visit = !document.cookie.match(/(?:^|; )__utma=(?:\d*\.)
{5}\d*/)
    ;
    if(!!custom_var && custom_var === 'Neuer Besucher auf Fanpage-Tab')
{
      _gaq.push(['_setCustomVar', 5,'Facebook-Status','Erst Fanpage-Tab
jetzt Website',1])
    }else if(first_visit && ref_is_fb){
      _gaq.push(['_setCustomVar', 5,'Facebook-Status','First Visit
durch Facebook',1])
    }else if(!custom_var){
      FB.getLoginStatus(function(response){
        if (response.status === 'connected' || response.status ===
'not_authorized') {
          _gaq.push(['_setCustomVar', 5,'Facebook-Status','FB-App
Nutzer',1])
        }
      });
    }
  });
}())
```

Listing 6.9: Integration der Benutzeranteile in die Fanpage-Messung

Erklärung der Änderungen:

```
,   ref_is_fb = !!document.referrer.match(/(?:.*)?\.facebook\.com(?:\/|
$)/)
,   first_visit = !document.cookie.match(/(?:^|; )__utma=(?:\d*\.)
{5}\d*/)
```

Es werden die Variablen `ref_is_fb` und `first_visit` definiert. Dabei wird jeweils das Ergebnis eines regulären Ausdrucks zu einem Wahr-Falsch-Wert gemacht und entsprechend in den Variablen gespeichert. Dabei kann `ref_is_fb` Auskunft darüber geben, ob ein Besucher von Facebook kam, und `first_visit`, ob dies der erste Besuch dieses Besuchers auf unserer Website ist.

```
else if(first_visit && ref_is_fb){
    _gaq.push(['_setCustomVar',  5,'Facebook-Status','First  Visit  durch
Facebook',1])
}
```

Ist der Wert für `first_visit` und der Wert für `ref_is_fb` wahr, kommt der Nutzer also von Facebook und ist das erste Mal auf unserer Website, wird die Custom-Variable auf den Wert *First Visit durch Facebook* gesetzt.

```
else if(!custom_var){
  FB.getLoginStatus(function(response){
    ...
  });
}
```

Ist keiner der bisherigen Fälle eingetreten, was durchaus wahrscheinlich ist, und die Custom-Variable noch nicht gesetzt, kommt der Part, an dem Facebook über den Login-Status des Nutzers gefragt wird.

Dazu wird die Methode *getLoginStatus* aufgerufen und dieser eine Callback-Funktion übergeben, die ein `response`-Objekt als Parameter erwartet.

```
if (response.status === 'connected' || response.status ===
'not_authorized') {
  _gaq.push(['_setCustomVar', 5,'Facebook-Status','Facebook-Nutzer',1])
}
```

Ist das Attribut *status* des `response`-Objekts auf `connected` oder `not_authorized` gesetzt, bedeutet dies, der Besucher ist bei Facebook angemeldet und erhält entsprechend den Wert *Facebook-Nutzer* in seiner benutzerdefinierten Variablen.

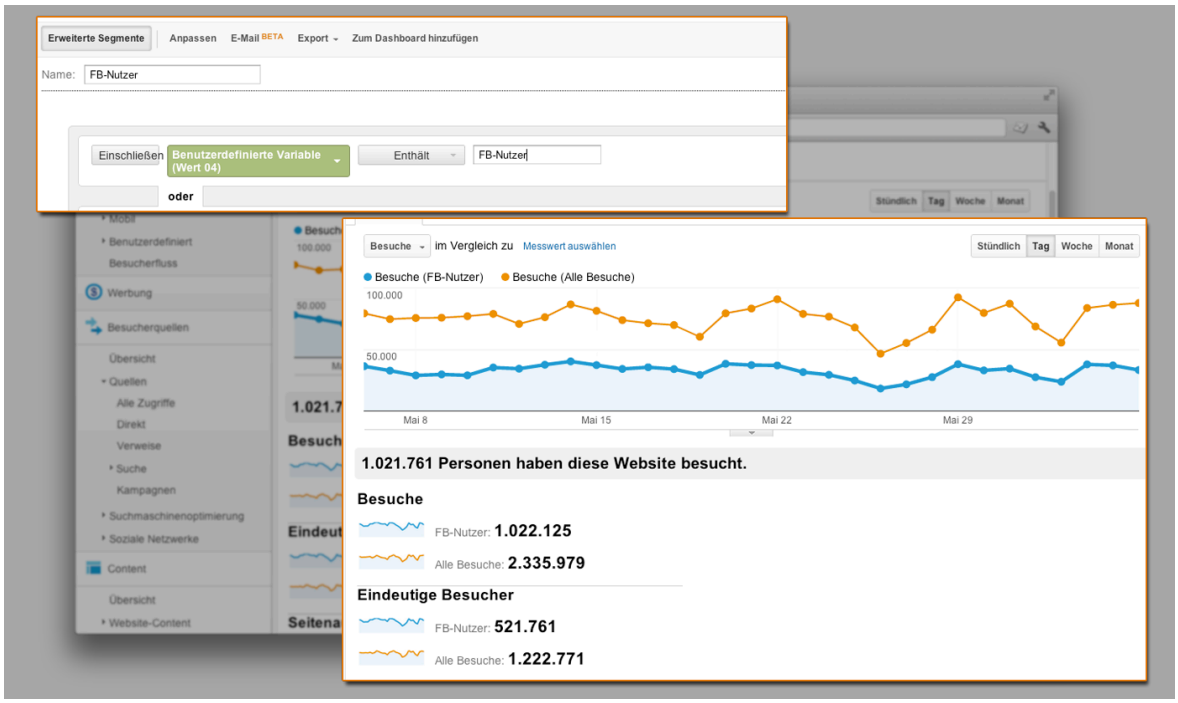



Abbildung 6.22: Facebook-Nutzeranteile mit benutzerdefinierten Variablen und erweiterten Segmenten

## Kampagnen für Facebook-Traffic

Neben den Links, die Sie selbst auf Facebook posten und entsprechend mit Kampagnen-Parametern versehen können, gibt es eine Vielzahl weiterer Möglichkeiten, wie Besucher über Facebook auf Ihre Website kommen können. Dies kann passieren, weil ein Nutzer einen Link zu Ihrer Website postet oder einen Beitrag auf Ihrer Website liked. Damit diese Besucher nicht in der Masse von anderem Referral-Traffic untergehen, können Sie diese Besucher einer Facebook-Kampagne zuweisen, um dies später gezielt auszuwerten.

URL-Parameter sind der derzeit einzige Weg, Besucher automatisch einer Kampagne zuzuweisen. Da Sie jedoch nicht sämtliche Links beeinflussen können,

die von Facebook zurück auf Ihre Website gehen, würden hier große Lücken in Ihrem Tracking entstehen.

Aus diesem Grund benötigen Sie erneut ein kleines Script, das dies automatisch übernimmt. Dieses Script muss die folgenden Dinge prüfen:

- Ist der Referrer des Besuchers eine Domain von Facebook, kommt der Besucher also von Facebook?
- Befindet sich in der URL kein anderer Kampagnen-Tracking-Parameter? Schließlich soll dieses Kampagnen-Tracking nur dann greifen, wenn der Link nicht bereits mit Kampagnen-Parametern versehen ist.
- Ist ein `fb_ref`-Parameter in der URL vorhanden? Wenn dies der Fall ist und die `ref`-Attribute entsprechend befüllt worden sind, kann dies genutzt werden, um den Besucher direkt dem entsprechenden Social-Plugin zuzuweisen.

Ein Tracking-Code, der diese Aufgaben übernimmt, sieht wie folgt aus:

```
;(function(){
  window._gaq = window._gaq || []
  var no_campaign = !document.location.href.match(/(?:\?|
&)utm_source=/)
  ,    ref_is_fb = !!document.referrer.match(/(?:.*)?(?:\.|\/)facebook
\.com(?:\/|$)/)
  ,    medium = (document.location.href.match(/(?:\?|&)(?:fb_ref\=)([^
\&]*)?(?:$|\&)/)||[,'organic'])[1]
  ,    campaign_params =
'utm_campaign=(organic)&utm_source=facebook&utm_medium='+medium
  ;
  if(no_campaign && ref_is_fb){
    _gaq.push(['_set', 'campaignParams', campaign_params]);
  }
}())
```

Listing 6.10: Kampagnen-Tracking für Facebook-Traffic

Erklärung des Codes:

```
var _gaq = _gaq || []
```

Wie jedes Mal wird zunächst das `_gaq`-Objekt initialisiert.

```
no_campaign = !document.location.href.match(/(?:\?|&)utm_source=/)
ref_is_fb = !!document.referrer.match(/(?:.*)?(?:\.|\/)facebook\.com(?:
\/|$)/)
```

Der Variablen `no_campaign` wird ein Wahr-Falsch-Wert zugewiesen, der über einen regulären Ausdruck erstellt wurde, der prüft, ob sich auf der derzeitigen Seite Kampagnen-Parameter befinden.

Der `ref_is_fb`-Variablen wird ebenfalls ein Wahr-Falsch-Wert zugewiesen, der klärt, ob es sich beim Referrer um Facebook handelt.

```
medium = (document.location.href.match(/(?:\?|\&)(?:fb_ref\=)([^\&]*)?
(?:$|\&)/)||[,'organic'])[1]
```

Das Medium für die Kampagne wird bestimmt. Ist das `fb_ref`-Attribut vorhanden, so wird dieser Wert aus der URL extrahiert und gespeichert. Ist der `fb_ref`-Parameter nicht gesetzt, wird stattdessen als Medium `organic` gesetzt.

```
campaign_params =
'utm_campaign=(organic)&utm_source=facebook&utm_medium=organic'
```

Der `campaign_params`-Variablen werden die Kampagnen-Parameter sowie das Medium zugewiesen.

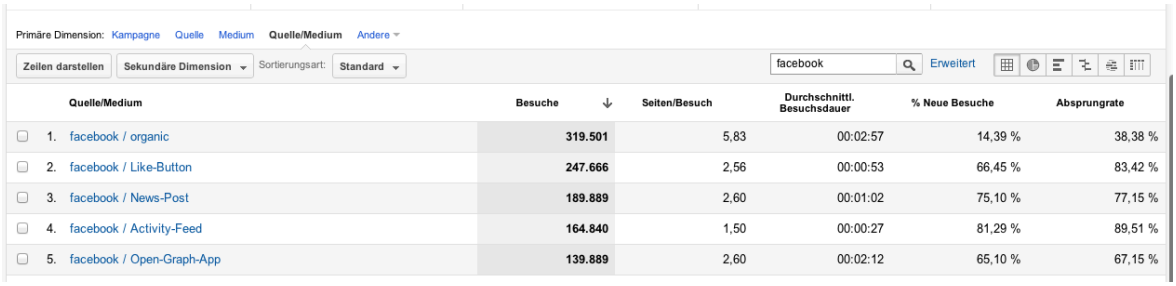
```
if(no_campaign && ref_is_fb){
  ...
}
```

Es wird geprüft, ob keine Kampagnen-Parameter existieren, der Referrer aber Facebook ist. Ist beides gegeben, wird der Code im `if`-Block ausgeführt.

```
_gaq.push(['_set', 'campaignParams', campaign_params]);
```

War die vorherige Anweisung korrekt, wird dem `_gaq`-Objekt mitgeteilt, die Kampagnen-Parameter auf die entsprechend vorher festgelegten Werte zu setzen.

Von jetzt an können Sie den organisch in Facebook entstandenen Traffic als eigene Kampagne in der Benutzeroberfläche von Google Analytics betrachten und auswerten.




| | Quelle/Medium | Besuche | Seiten/Besuch | Durchschnittl. Besuchsdauer | % Neue Besuche | Absprungrate |
|---|---|---|---|---|---|---|
| 1. | facebook / organic | 319.501 | 5,83 | 00:02:57 | 14,39 % | 38,38 % |
| 2. | facebook / Like-Button | 247.666 | 2,56 | 00:00:53 | 66,45 % | 83,42 % |
| 3. | facebook / News-Post | 189.889 | 2,60 | 00:01:02 | 75,10 % | 77,15 % |
| 4. | facebook / Activity-Feed | 164.840 | 1,50 | 00:00:27 | 81,29 % | 89,51 % |
| 5. | facebook / Open-Graph-App | 139.889 | 2,60 | 00:02:12 | 65,10 % | 67,15 % |

Abbildung 6.23: Übersicht verschiedener Kampagnen für Facebook-Traffic

## Facebook Fanpages Plus

von Tim Sebastian


**Kaufen:**
mitp.de
Amazon.de

24,95€
ISBN 978-3826691843
2012
320 Seiten

Dieses Buch zeigt detailliert, wie eine Facebook Fanpage grundlegend aufgebaut und erweitert wird. Der Autor zeigt, wie Fanpage Applikationen erstellt werden wie zum Beispiel Tab Apps und Fan Gating. Des Weiteren erläutert er, wie das Open Graph Protokoll und Social Plug-ins eingesetzt werden. Abschließend geht er auf das Tracking ein.

Mehr Informationen...

# allfacebook.de

Der inoffizielle Facebook Blog

Jetzt Fan werden:

facebook.com/marketingde

Philipp Roth & Jens Wiese

kontakt@allfacebook.de