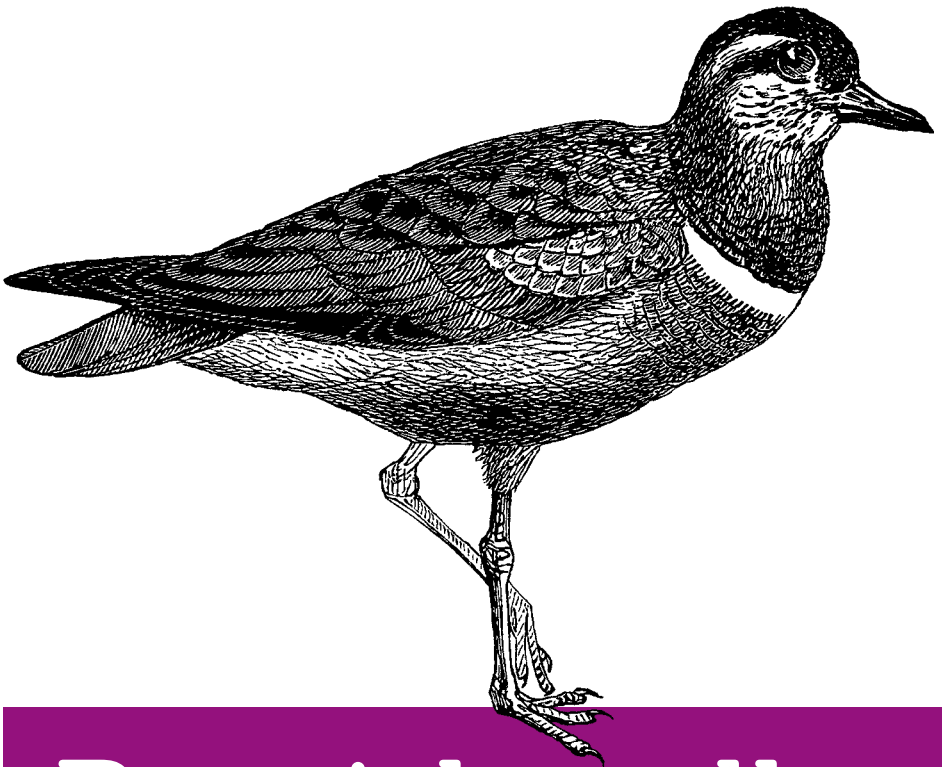
O'REILLY®



Praxishandbuch Facebook-Programmierung

FACEBOOK-ANWENDUNGEN MIT JAVASCRIPT UND PHP

Stephan Alber,
Klaus Breyer, Kornelius Nägele

1. AUFLAGE

Praxishandbuch Facebook-Programmierung

Stephan Alber
Klaus Breyer
Kornelius Nägele

O'REILLY®
Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

Kommentare und Fragen können Sie gerne an uns richten:
O'Reilly Verlag
Balthasarstr. 81
50670 Köln
E-Mail: kommentar@oreilly.de

1. Auflage 2015

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.ddb.de* abrufbar.

Lektorat: Volker Bombien, Köln
Korrektorat: Dr. Dorothée Leidig
Satz: III-satz, Husby, www.drei-satz.de
Umschlaggestaltung: Michael Oreal, Köln
Produktion: Andrea Miß, Köln
Belichtung, Druck und buchbinderische Verarbeitung:
Druckerei Kösel, Altusried-Krugzell

ISBN 978-3-95561-794-3

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

Inhalt

KAPITEL 10

Anwendungsoptimierung und Erfolgsmessung

Der altbewährte Spruch »Vertrauen ist gut, Kontrolle ist besser« findet auch in der Facebook-Anwendungsentwicklung Anwendung: Ob eine Anwendung durch ein Update wirklich verbessert wurde, sollte nicht nur dem Bauchgefühl überlassen werden. Gleiches gilt für eine Marketing-Kampagne: Fragen wie »Warum haben so wenige Nutzer das Gewinnspielformular letztendlich abgesendet?« sollten mit Messungen und Analysen und nicht Vermutungen beantwortet werden. Ebenso wichtig ist die Herkunft der Nutzer: Wenn 50% der Visits aus dem Ausland kommen, war die Marketing-Aktion für den lokalen Bioladen nur halb so erfolgreich wie gedacht.

Als Analyse-Tools werden wir zwei Services behandeln: *Facebook Insights* und *Google Analytics*. Beide Tools haben ihre Spezialgebiete. Alle mit Facebook-API verwandten Statistiken finden sich erwartungsgemäß in Facebook Insights. Neben Nutzerstatistiken werden von Facebook Verweise auf Anwendungen aufgezeichnet. Google Analytics eignet sich besonders für klassische »Website«-Fragen. Woher kommen meine Besucher, wie viele Seiten werden pro Tag abgerufen, welche Browser verwenden die Nutzer?

Facebook Insights

Im Januar 2014 veröffentlichte Facebook die Version 2.0 des Anwendungsstatistik-Tools »Insights«. Die neue Version hat den Beta-Status verlassen, es sind dennoch weiterhin Updates zu erwarten. Alle Daten reichen lediglich vier Monate zurück.

Eine Übersicht wichtiger Informationen, wie die monatlich aktiven Nutzer, wird im Dashboard jeder Anwendung angezeigt. Ausführliche Zahlen sind unter dem Punkt *Insights* bzw. *Statistiken* (linke Navigationsspalte) beheimatet. Alternativ finden sich Links zu allen Anwendungsstatistiken unter *https://www.facebook.com/insights/* (siehe Abbildung 10-1).

Metrik: Referrals

»Woher kommen meine Nutzer?« ist eine der am häufigsten gestellten Fragen im Bereich der Anwendungsstatistiken. Die Antwort(en) hierfür finden sich im Reiter *Referrals*. Facebook gibt allerdings nur Aufschluss über die Herkunft von Nutzern innerhalb der Facebook-Plattform. Für externe Referrals muss ein Tool wie Google Analytics verwendet werden (siehe Abbildung 10-2).

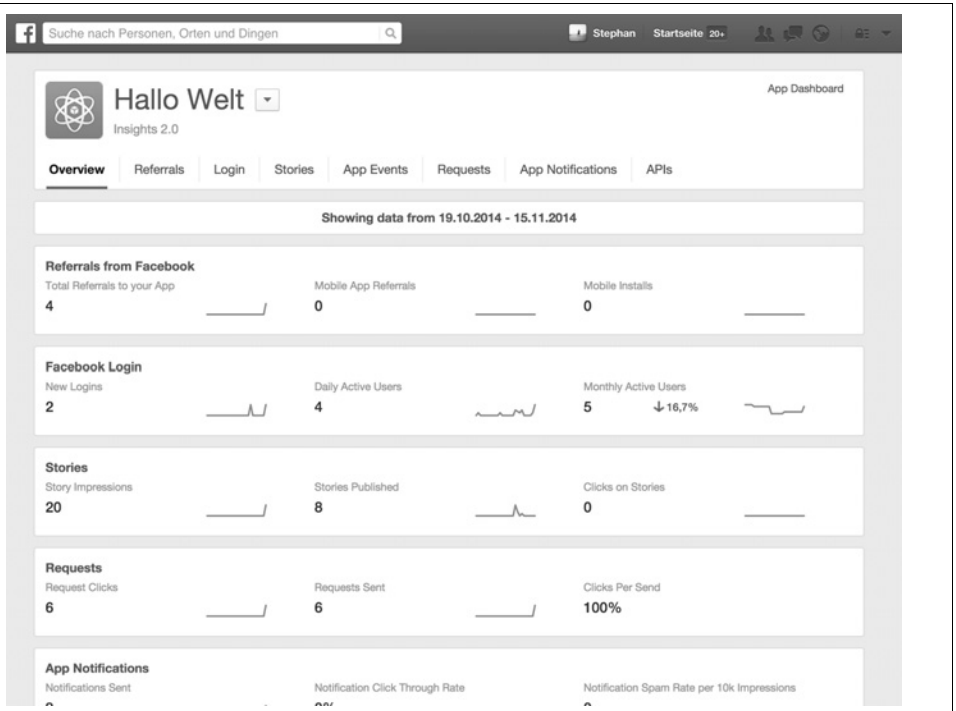


Abbildung 10-1: Das Insights-Dashboard gibt eine Zusammenfassung aller Messbereiche

Abbildung 10-2: Wichtige Informationen zum Wachstum der Anwendung – die »Referrals« zur Anwendung

Metrik: Facebook-Login

Die wichtigste Zahl für jeden Entwickler ist die Anzahl der Nutzer seiner Anwendung. Facebook teilt sowohl relative als auch absolute Zahlen mit. Zur Performance-Erfassung werden Neuanmeldungen, die Gesamtzahl der Nutzer sowie täglich bzw. monatlich aktive Nutzer publiziert.

Als aktive Nutzer werden alle Nutzer bezeichnet, die die Anwendung installiert und einen der möglichen Kanäle – Canvas, Seitenreiter oder Website – eingeloggt verwendet haben. Zahlen werden in täglichen, wöchentlichen und monatlichen Perioden ermittelt. Dabei ist zu beachten, dass insbesondere die Zahl der monatlich aktiven Nutzer Hochrechnungen beinhaltet, d. h., wenn eine neue App in der ersten Woche 100 Nutzer gewinnen konnte, geht Facebook von einer monatlichen Nutzerzahl von beispielsweise 400 aus. Mit längerer Laufzeit werden diese Zahlen genauer, es sind dennoch Rundungsfehler und Schätzwerte weiterhin möglich.

In der Folge versteht sich: Nicht-eingeloggte Nutzer werden hierbei nicht erfasst. Wie ein Überblick über das Gäste-Nutzer-Verhältnis erstellt werden kann, erklären wir im folgenden Google-Analytics-Teil unter *Nicht-eingeloggte und eingeloggte Nutzer erfassen*.

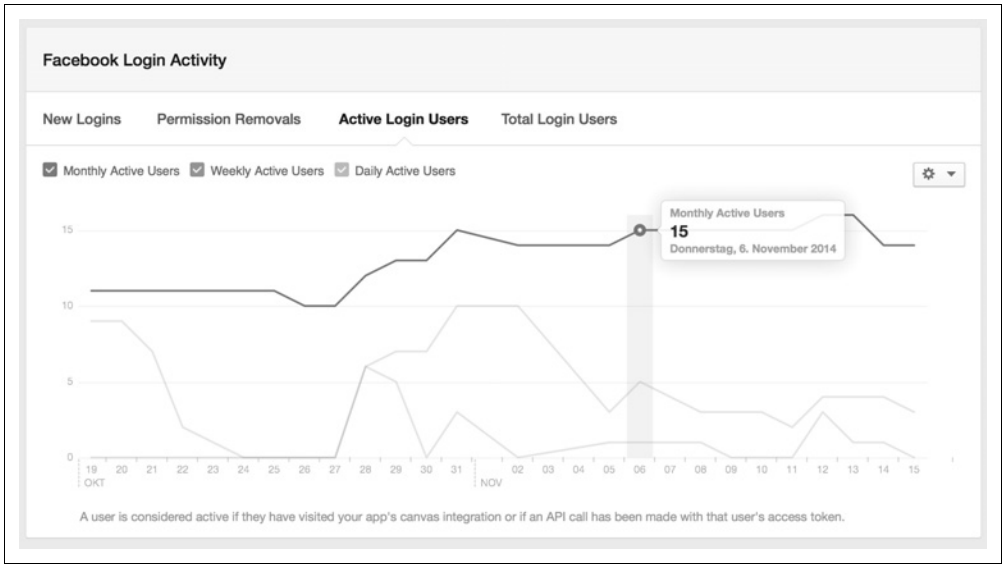


Abbildung 10-3: Statistiken über das Login-Verhalten der Nutzer einer Anwendung

Metrik: Storys

Die Metrik Storys informiert über den Erfolg deiner Open-Graph-Storys, Likes und Kommentare. Diese Zahlen sind von äußerster Wichtigkeit, da es sich hierbei um mehr oder weniger kostenlose Werbeanzeigen handelt. Sind die Inhalte bzw. die Beiträge deiner Anwendung interessant, können Tausende oder Millionen Nutzer ohne Werbeausgaben erreicht werden.

Die primären Messwerte dieser Metrik sind *Clicks* – zu sehen in Abbildung 10-4 – sowie *Impressions*, d.h. die Auskunft, von wie vielen Nutzern die Beiträge gesehen wurden, zu sehen in Abbildung 10-5.

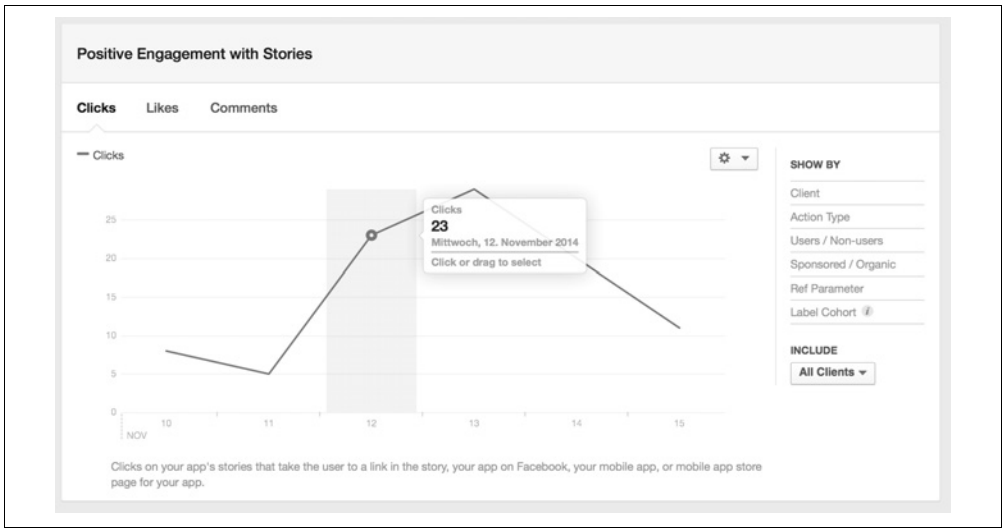


Abbildung 10-4: Die Metrik Storys informiert über die Reichweite von Beiträgen

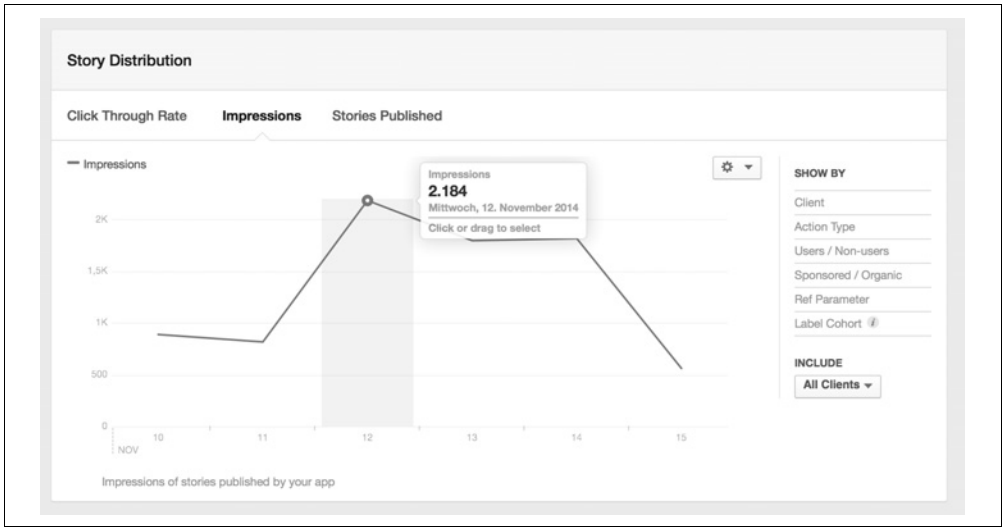


Abbildung 10-5: Anzahl der Impressions einer Story

Weitere Metriken

Darüber hinaus bietet Facebook Insights weitere Metriken über Anfragen (Requests), Anwendungsbenachrichtigungen (App Notifications) und API-Anfragen. Bei den ersten

beiden Metriken sollten insbesondere die Punkte »Negatives Feedback« sowie »Negatives Engagement« beachtet werden.

App-Events für Canvas-Anwendungen

Facebook Insights erfassen ohne weitere Einrichtung eine Vielzahl Ereignisse, die auf der Facebook-Plattform stattfinden. Wie besprochen, gehört hierzu zum Beispiel die Verbreitung von Open-Graph-Storys. Als neues Feature der Insights-Version 2.0 lassen sich darüber hinaus auch gezielt Ereignisse (*Events*) in der Anwendung erfassen.

Das Event-Tracking findet über das Facebook-JavaScript-SDK statt. Als Ereignis kann entweder ein von Facebook vordefiniertes Ereignis oder ein benutzerdefiniertes Ereignis gesendet werden. Dazu können den Ereignissen Eigenschaften zugeordnet werden. Beim Erfassen einer Suchanfrage wäre dies beispielsweise der Suchtext.

Besonderen Ertrag liefert das Event-Tracking für native mobile Anwendungen (iOS und Android). Hierbei werden zusätzlich zu den erfassten Ereignissen Informationen wie die Version der Anwendung oder das Gerätemodell erfasst. Des Weiteren ist eine Verknüpfung mit Anzeigen zum Analysieren von Anwendungsinstallationen möglich.

Der Großteil der Funktionen kann auch über das JavaScript-SDK verwendet werden, allerdings mit der Einschränkung, dass die Tracking-Funktionen nur in Canvas-Anwendungen verfügbar sind.

Demographische Aufschlüsselung

Ein besonderer Zusatz der App-Events-Statistiken ist die demographische Aufschlüsselung der Nutzer für jedes Ereignis. Jede Ansicht kann nach Alter, Geschlecht, Sprache und Herkunftsland gefiltert werden. Dies erlaubt Analysen darüber, in welchen Nutzerkreisen einzelne Funktionen mehr oder weniger gut ankommen.

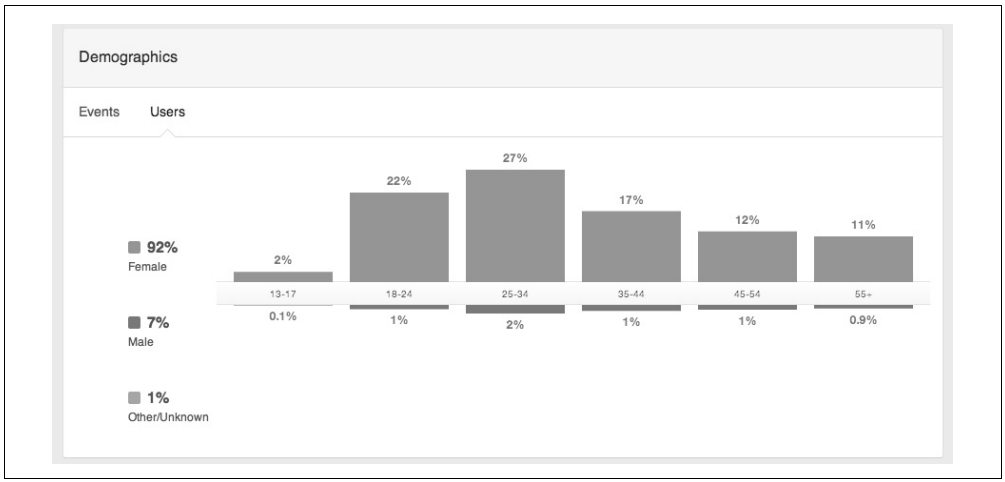


Abbildung 10-6: Demographische Aufschlüsselung der Nutzer eines registrierten Ereignisses

Technische Implementierung

Über das JavaScript-SDK können drei unterschiedliche Grundereignisse erfasst werden: das Aktivieren bzw. Laden der Anwendung, ein allgemeines Ereignis oder ein Einkauf:

```
FB.AppEvents.activateApp();
FB.AppEvents.logEvent(...);
FB.AppEvents.logPurchase(...);
```

Über allgemeine Ereignisse können die von Facebook vordefinierten Ereignisse sowie benutzerdefinierte Ereignisse festgehalten werden.

Laden der Anwendung

Die Statistik über das Laden einer Anwendung ist offensichtlich eine Metrik, die aus der mobilen Anwendungsentwicklung entstanden ist. Im Gegensatz zu mobilen Anwendungen gibt es in Webanwendungen ein Verhalten wie das Minimieren und Zurückkehren zu einer Anwendung nicht bzw. die Systemlimitierungen lassen es nicht zu, auf entsprechende Ereignisse zuzugreifen.

Um die Metrik dennoch verwenden zu können, sollten Kriterien erstellt werden, wann ein ähnliches Ereignis eintritt. Beispielsweise könnte dies der erste Seitenabruf sein, bei weiteren Seitenabrufen wird das erneute Tracking über ein Cookie unterbunden. Hier zeigt sich allerdings schnell ein Problem und gleichzeitig der deutliche Unterschied zu mobilen Anwendungen: Wie lange soll das Cookie erhalten bleiben, bis ein erneutes Laden der Anwendung registriert werden soll? Schließt der Nutzer die Anwendung und kehrt innerhalb der Laufzeit des Cookies wieder zurück, so wird dieser Vorgang nicht erfasst; werden keine Cookies verwendet, kommt es zu falschen Zahlen, wenn der Nutzer mehrere Browser-Fenster verwendet.

Dennoch, solltest du eine Lösung gefunden haben, kannst du das Laden der Anwendung über den einfachen Befehl

```
FB.AppEvents.activateApp();
```

erfassen. Der Aufruf erwartet keine weiteren Argumente.

Allgemeine Ereignisse

Universeller hingegen ist das Erfassen allgemeiner Ereignisse. Hierbei ist klar auf den Unterschied zwischen Webseite und Anwendung bzw. Spiel hinzuweisen: Während man sich in klassischen Webseiten primär auf Seitenabrufe, URLs, Inhaltsbereiche und Ähnliches konzentriert, sind bei Anwendungen einzelne Interaktionen von größerer Bedeutung.

Eine beispielhafte Problemstellung am Fall »Tippspiel«: Die meistbesuchte Seite wird höchstwahrscheinlich das Formular zur Tippabgabe sein. Der klassischen Messung

zufolge handelt es sich um eine gute Ansicht, da viele Nutzer die Seite besuchen. Was nicht erfasst wird, ist, ob die Nutzer das Formular überhaupt absendet, und wenn, ob das Senden erfolgreich war.

Über die Messung der Schritte bzw. Ereignisse *Formular abgerufen*, *Formular abgesendet*, *Fehler im Formular* und *Formular erfolgreich gesendet* kann gezielt analysiert werden, ob die hohe Anzahl der Abrufe durch eine gute Implementierung der Funktion oder durch »stecken gebliebene« Nutzer hervorgerufen wurde.

Das beschriebene Beispiel kann über ein benutzerdefiniertes Ereignis realisiert werden. Rufe dazu wie im folgenden Code-Snippet die Funktion `FB.AppEvents.logEvent()` auf und vergib als erstes Funktionsargument einen Namen für dein Ereignis. In unserem Fall ist dies `'bet_form_sent'` für das Absenden des Formulars.

Als zweites Argument kann ein numerischer Wert übergeben werden. Dieser Wert ist nur in bestimmten Fällen sinnvoll und ist daher optional. Ein Beispiel hierfür wäre das Einladen von Freunden. Dabei könnte erfasst werden, wie viele Freunde eingeladen wurden.

Im dritten Argument können bis zu 25 weitere Textwerte oder numerische Werte festgehalten werden. Sie können später als Filter verwendet werden. Die Übergabe erfolgt aus einem Key-Value-Hash.

```
var params = {
    games_submitted: 3,
    retries: 0
};
FB.AppEvents.logEvent(
    'bet_form_sent',
    null,
    params
);
```

Vordefinierte Ereignisse

Neben der Möglichkeit, benutzerdefinierte Ereignisse zu verwenden, verweist Facebook auf seine vordefinierten Ereignisse und empfiehlt ihre Verwendung. Nach aktuellem Stand ergeben diese Ereignistypen neben der Inspiration und Struktur keinen besonderen Mehrwert und dienen aller Wahrscheinlichkeit nach Facebook zur strukturierten Datenerfassung über alle Anwendungen hinweg.

Dies ist in erster Linie keine technische Frage, aber an diesem Funktionsbeispiel wird deutlich: Facebook ist kostenlos und es ist nicht kostenlos. Daten sind eine Währung. Wenn du in deiner Anwendung detaillierte Daten über Nutzerverhalten mit Hilfe des Facebook-Insights-Tools erfasst, musst du dir darüber im Klaren sein, dass Facebook diese Daten ebenfalls verwenden wird. Gleiches gilt allerdings auch für andere Messtools wie Google Analytics.

Doch zurück zu den technischen Einzelheiten: Zu den vordefinierten Ereignissen gehören das Erreichen eines Levels, Bewerten eines Inhalts, Suchanfragen, Abschluss eines Tutoriums, Abschluss der Registrierung oder Ansicht eines Inhalts. Alle verfügbaren Ereignisse verbinden sich in der Variable `FB.AppEvents.EventNames`, d.h., du kannst die Liste direkt in deiner Browser-Konsole über `console.log(FB.AppEvents.EventNames)` abrufen.

Selbstverständlich findet sich eine komplette Auflistung der Ereignisse auch in der Facebook-Developer-Dokumentation unter *https://developers.facebook.com/docs/canvas/appevents*. Bestimmte Ereignisse erwarten das Mitsenden entsprechender Parameter, sie finden sich in der Variable `FB.AppEvents.ParameterNames`.

Das Registrieren einer Suchanfrage zeigen wir im folgenden Code-Beispiel. Der Schlüssel für den Suchtext im Parameter-Hash wird über `FB.AppEvents.ParameterNames.SEARCH_STRING` festgelegt, der Name des Events über `FB.AppEvents.EventNames.SEARCHED`:

```
var params = {};
params[FB.AppEvents.ParameterNames.SEARCH_STRING] = 'Suchtext';
FB.AppEvents.logEvent(
    FB.AppEvents.EventNames.SEARCHED,
    null,
    params
);
```

Einkäufe

Zu guter Letzt kommt das Erfassen von Einkäufen. Auch wenn dies ein eher seltener Fall sein sollte – wenn er vorkommt, ist er umso wichtiger. Die Nutzung von Facebook-Payment erfordert die Registrierung einer Firma, die Definition von Produkten usw. Der Vorgang wird in der Facebook-Dokumentation ausführlich beschrieben (*https://developers.facebook.com/docs/payments*).

Da Einkäufe, die in Facebook-Canvas-Anwendungen über das Facebook-Payment-System ausgeführt werden, ohnehin automatisch erfasst werden, ist manuelles Tracking dieser Einkäufe nicht notwendig.

Die Funktion kann allerdings beim Verkauf von materiellen Gütern Sinn machen. Verwende in diesem Fall folgendes Format:

```
var params = {};
params[FB.AppEvents.ParameterNames.CONTENT_ID] = '978-3955617943';
FB.AppEvents.logPurchase(34.90, 'EUR', params);
```

Die Währungscodes entsprechen dem ISO-4217-Standard für Währungen (*http://en.wikipedia.org/wiki/ISO_4217*).

Soziale Interaktionen erfassen

Obwohl Facebook Insights bereits eine Reihe von Statistiken über Plattform-Interaktionen erstellt, können über App-Events noch weitere soziale Interaktionen erfasst werden.

Ein Anwendungsfall hierfür ist das Tracking eines Like-Buttons, wie er in Kapitel 4 *Open Graph und soziale Plugins* beschrieben wird:

```
<div class="fb-like" data-href="https://domain.com/" data-layout="button_count" data-
action="like" data-show-faces="false" data-share="true"></div>
```

Um neue Likes (bzw. Unlikes) zu messen, wird die Facebook-JavaScript-SDK-Funktion `FB.Event.subscribe()` verwendet. Bei einem Klick auf den Like-Button wird die Nutzer-Interaktion an Facebook Insights gesendet:

```
FB.Event.subscribe('edge.create', function(targetUrl) {
    var params = {
        url: targetUrl
    };
    FB.AppEvents.logEvent(
        'liked',
        null,
        params
    );
});
FB.Event.subscribe('edge.remove', function(targetUrl) {
    var params = {
        url: targetUrl
    };
    FB.AppEvents.logEvent(
        'liked',
        null,
        params
    );
});
```

Neben dem Like-Button ist es möglich, das Erstellen und Löschen von Kommentaren mittels des sozialen Plugins *Comments* (`comment.create`, `comment.remove`), die Verwendung des Send-Buttons (`message.send`) sowie das Ein- und Ausloggen zu bzw. von Facebook (`auth.login`, `auth.logout`) zu erfassen.

Testen der Implementierung

Ergebnisse deiner Messungen erscheinen erst nach ca. 24 Stunden in der Insights-Ansicht. Dies erschwert die Entwicklung natürlich etwas. Doch die Implementierung kann unmittelbar über die »*Most recently logged events*«-Ansicht getestet werden. Du erreichst sie über den Button »Show Most Recent Events« im Reiter »*App Events*«.

Es werden maximal 20 Ereignisse zwischengespeichert, des Weiteren besteht eine Verzögerung von 3 Sekunden. Warte daher einen Moment nach dem Absenden des Tests, um eine unnötige Fehlersuche zu vermeiden. Ein Protokoll unserer Tests zeigen wir in Abbildung 10-7.

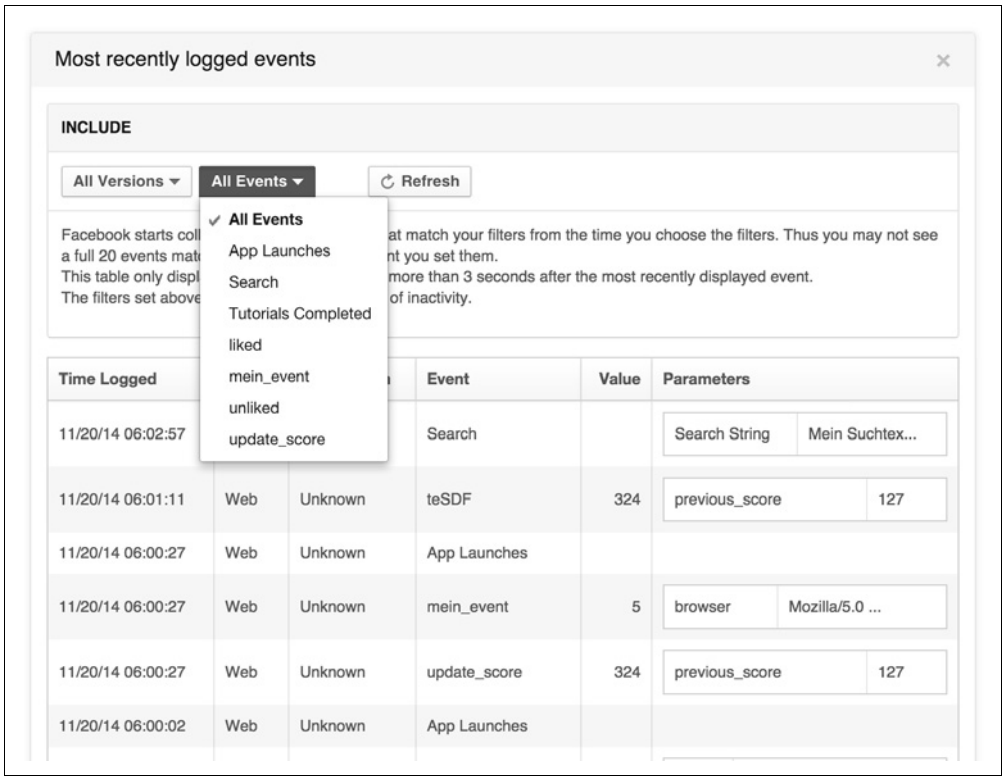


Abbildung 10-7: Die Implementierung des Tracking-Codes kann über die Ansicht »Most recently logged events« getestet werden

Tracking mit Google Analytics

Auf die Erstellung des (kostenlosen) Accounts möchten wir an dieser Stelle nicht weiter eingehen. Alle Informationen hierzu finden sich im Google-Analytics-Portal (*http://www.google.com/analytics/*). Google Analytics kann ohne Konflikte in jede Facebook-Anwendung eingebaut werden. Hierzu einfach das von Google angebotene Snippet 1:1 in das HTML-Template einbinden. Ob nach oder vor dem Laden des Facebook-JavaScript-SDKs ist nicht von Belang.

Neben den bereits zuvor erwähnten Statistiken lohnt sich für Facebook-Projekte besonders ein Blick in den Bereich *»Akquisition → Soziale Netzwerke«*. Hier wird detailliert dargestellt, über welche sozialen Netzwerke Besucher auf die Anwendung bzw. Website gelangen. Canvas- sowie Seitenreiter-Anwendungen sind hier im Nachteil: Durch die Verwendung eines Iframe zur Darstellung der Anwendung geht die Referrer-Information verloren.

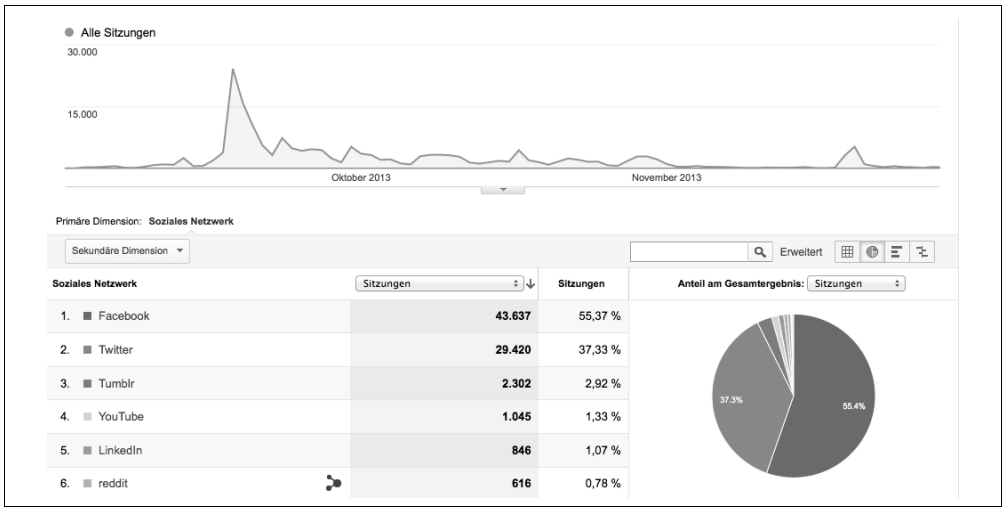


Abbildung 10-8: Über Google Analytics können Einflüsse aller sozialen Netzwerke gemessen werden

Referrer-Tracking-Fix: Canvas-Apps

Für eigene bzw. kontrollierte Promotion-Aktivitäten und Link-Sharing kann dieses Problem mit ein paar Anpassungen umgangen werden. Hierzu ist es erforderlich, dass jeder Link mit einen Referrer-Parameter erweitert wird, z. B. anstelle des Links

```
https://apps.facebook.com/app-name/
```

muss ein Link nach dem Schema

```
https://apps.facebook.com/app-name/?fbRef=https://www.facebook.com/marketing
```

geteilt werden. Kleiner Tipp, um die URL im Timeline-Beitrag nicht zu lang erscheinen zu lassen: URL Shortner Services wie Bitly (*https://bitly.com/*) schaffen Abhilfe.

Der mitgesendete Parameter wird per JavaScript ausgelesen und zum Überschreiben des Referrer-Wertes verwendet:

```
function getQueryVariable(variable) {
    var query = window.location.search.substring(1);
    var vars = query.split('&');
    for (var i = 0; i < vars.length; i++) {
     var pair = vars[i].split('=');
     if (decodeURIComponent(pair[0]) == variable) {
            return decodeURIComponent(pair[1]);
     }
     }
}
var ref = getQueryVariable('fbRef');
if(ref) {
  ga('set', 'referrer', ref);
}
ga('send', 'pageview');
```

Falls ein Wert gesetzt wurde, wird mittels `ga('set', 'referrer', ref)` der Referrer gesetzt. Es ist darauf zu achten, dass das Überschreiben des Wertes vor dem Senden des Page-Views – `ga('send', 'pageview')` – durchgeführt wird. Zur Überprüfung der Funktionalität kann die Google-Chrome-Extension »*Google Analytics Debugger*« eingesetzt werden (kostenlos im Google-Chrome-Webstore, *https://chrome.google.com/webstore/*). Des Weiteren wird der Testaufruf in der Google-Analytics-»Echtzeit«-Ansicht unter »Häufigste soziale Zugriffe« zu sehen sein.

Referrer-Tracking-Fix: Seitenreiter

Etwas komplizierter sieht es bei den Seitenreiter-Apps aus. In diesem Fall kommt erneut der in Kapitel 3 behandelte Parameter `app_data` zum Einsatz. Als Parameter wird derselbe Wert wie zuvor verwendet, diesmal jedoch *URL-Encoded*, d. h., aus dem Pärchen

```
fbRef=https://www.facebook.com/marketing
```

wird

```
fbRef%3Dhttps%3A%2F%2Fwww.facebook.com%2Fmarketing
```

Werden die Argumente nicht URL-encoded, geht der Referrer-Wert verloren, da er als neue Zeichenkette angesehen wird. Um ein Key-Value-Paar umzuwandeln, kann entweder die PHP-Funktion `urlencode()` (*http://us3.php.net/urlencode*) oder beispielsweise ein Online-Tool wie Meyerwebs URL-Decoder/-Encoder (*http://meyerweb.com/eric/tools/dencoder/*) verwendet werden.

Eine vollständige URL mit Facebook-Referrer-Argument setzt sich aus der Seiten-URL, der Seitenreiter-ID und dem `app_data`-Argument zusammen:

```
https://www.facebook.com/pages/My-Hello-World-Page/?sk=app_{APP_ID}&app_
data=fbRef%3Dhttps%3A%2F%2Fwww.facebook.com%2Fmarketing
```

Da das `app_data`-Argument im Signed Request übergeben wird, muss es per PHP ausgelesen werden:

```
require 'vendor/autoload.php';
use Facebook\Entities\SignedRequest;
$signed_request = SignedRequest::parse(
  $_POST['signed_request'], null, '{app-secret}'
);
$ref = false;
if(isset($signed_request['app_data'])) {
  parse_str($signed_request['app_data'], $app_data);
  $ref = isset($app_data['fbRef']) && $app_data['fbRef'] ?
            $app_data['fbRef'] : '';
}
```

Anschließend wird der Referrer analog zur JavaScript-Variante überschrieben:

```
<?php if($ref): ?>
    ga('set', 'referrer', '<?php echo $ref; ?>');
<?php endif; ?>
```

Soziale Interaktionen erfassen

Google Analytics hat seit einiger Zeit die Erfassung von sozialen Interaktionen im Programm. Im Gegensatz zur Facebook-Variante ist hier selbstverständlich die Zusammenführung verschiedener Netzwerke besonders interessant.

Wie in unserem Facebook-Insights-Beispiel können wir neue Likes sowie »Unlikes« messen. Bei einem Klick auf den Like-Button wird die Nutzer-Interaktion an Google Analytics gesendet:

```
FB.Event.subscribe('edge.create', function(targetUrl) {
  _gaq.push(['_trackSocial', 'facebook', 'like', targetUrl]);
});
FB.Event.subscribe('edge.remove', function(targetUrl) {
  _gaq.push(['_trackSocial', 'facebook', 'unlike', targetUrl]);
});
```

Dieselbe Vorgehensweise können wir auf die Kommentare, den Send-Button und den Facebook-Login anwenden. Die Ergebnisse dieser Messungen finden sich unter *Akquisition → Soziale Netzwerke → Plugins*.

Individuelle Interaktionen erfassen

Das Tracking von (sozialen) Interaktionen ist nicht an diese Standardbeispiele gekoppelt. So wäre es möglich, das Tracking zum Button-Klick in der Voting-App aus Kapitel 7 hinzuzufügen:

```
_gaq.push(['_trackSocial', 'facebook', 'vote', '{bild-url}']);
```

Allgemein folgt das Tracking diesem Schema:

```
_gaq.push(['_trackSocial', network, socialAction, opt_target, opt_pagePath]);
```

Das erste Argument `network` bezeichnet das aktive soziale Netzwerk, neben Facebook kann dies auch Twitter oder LinkedIn sein. Die `socialAction` repräsentiert die ausgeführte Nutzeraktion (z.B. »Share«). Beinhaltet die Interaktion ein Objekt, das eine eindeutige URL besitzt, kann diese Information per `opt_target` übergeben werden. Der ebenfalls optionale Parameter `opt_pagePath` wird von Google Analytics automatisch mit der aktuellen URL befüllt, kann allerdings, falls gewünscht, überschrieben werden. Weitere Informationen zum Social Tracking gibt es in der Google-Analytics-Dokumentation unter der URL *https://developers.google.com/analytics/devguides/collection/gajs/gaTrackingSocial*.

Nicht-eingeloggte und eingeloggte Nutzer erfassen

Um das Verhältnis zwischen nicht-eingeloggten und eingeloggten Nutzern zu erfassen, muss im Google-Analytics-Profil eine neue Messgröße (»Custom Dimensions«) erstellt werden. Geh hierzu zum Punkt *Verwalten → Property → Benutzerdefinierte Definitionen → Benutzerdefinierte Dimensionen* und füge eine neue Dimension mit dem Namen *guest* und dem Umfang *Hit* hinzu.

Da dieser Wert unmittelbar beim Seitenaufruf gesetzt werden soll, bietet sich die Nutzung des PHP-SDKs an. Wahlweise kann das JavaScript-SDK verwendet werden. Dies verzögert allerdings das Pageview-Tracking, da es etwas Zeit in Anspruch nimmt, den Login-Status eines Facebook-Nutzers zu erhalten. Im schlimmsten Fall können Pageviews durch die Ladeverzögerung oder Session-Fehler komplett verloren gehen.

Zur Implementation: Zunächst muss sichergestellt werden, dass über einen der Facebook-Login-Helper die Session überprüft wird:

```
$helper = new FacebookJavaScriptLoginHelper();
$session = $helper->getSession();
```

Im JavaScript-Teil, vor der Ausführung von `ga('send', 'pageview')`, wird die benutzerdefinierte Dimension gesetzt:

```
ga('set', 'guest', <?php echo intval(!$session): ?>);
```

Ist eine Session vorhanden, wird der Wert auf `0` gesetzt. Dieser kleine Trick ermöglicht es, sämtliche Statistiken nach diesem Faktor zu separieren. Zur Darstellung der Daten müssen lediglich zwei benutzerdefinierte Segmente erstellt werden. Der Button hierfür (*Neues Segment erstellen*) ist über den ^-Button, der links oben neben dem Reiter »Alle Sitzungen«-Segment Labels in jeder Statistik angezeigt wird, erreichbar.

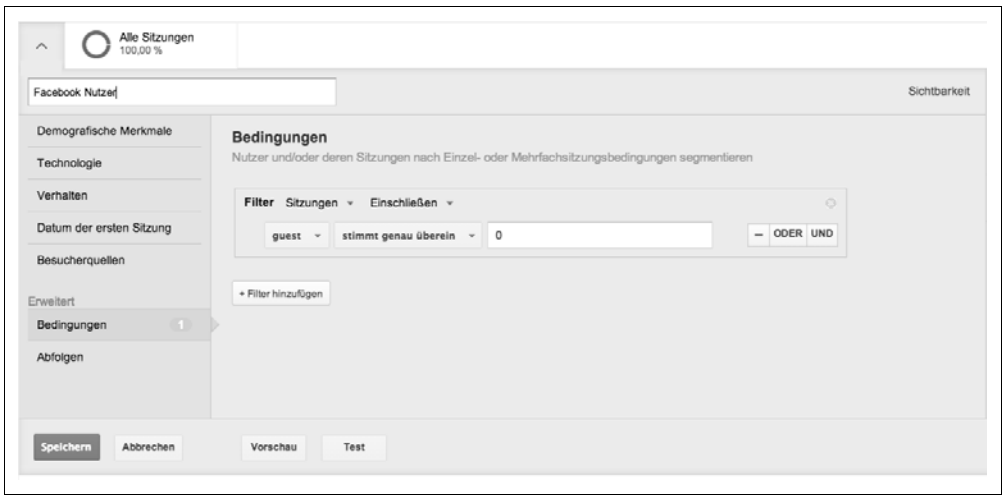


Abbildung 10-9: Über benutzerdefinierte Segmente kann Besucher-Traffic genauestens untersucht werden

Über das Menu *Bedingungen* lässt sich ein benutzerdefinierter Filter erstellen. Die Bedingung ist »*guest*« und »*stimmt genau überein*« mit `0`. Der Kehrwert dieser Bedingung (`1`) ergibt den Filter für das *Gäste*-Segment. Die Segmente lassen sich per Drag & Drop in jeder Statistikansicht anwenden. Im Beispiel ist die Gesamtzahl der Sitzungen zu sehen. Von besonderem Interesse sollte die *Aufschlüsselung nach Content* (unter *Verhalten* → *Website-Content*) sein. Diese Kombination dient als Indikator, welche Inhalte bei Gästen bzw. angemeldeten Nutzern mehr oder weniger gut ankommen.

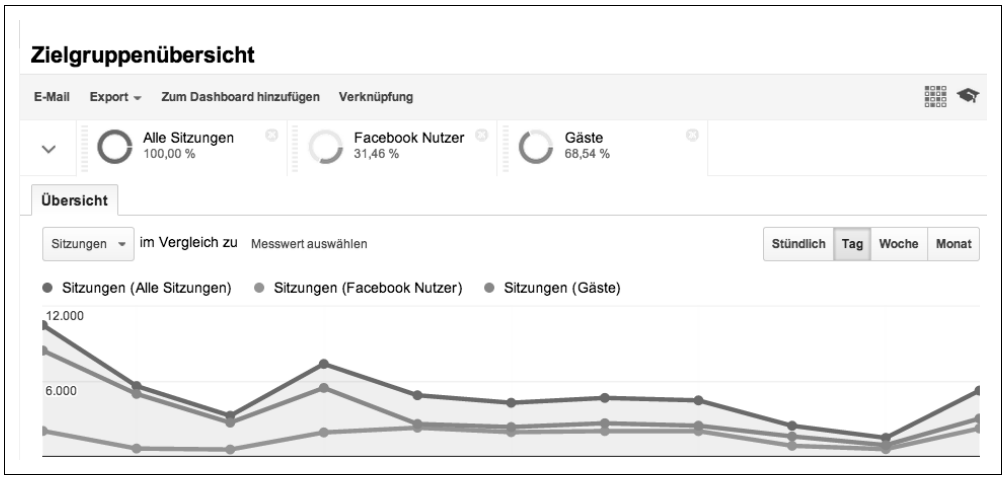


Abbildung 10-10: Die Entwicklung des Verhältnisses von Facebook-Nutzern und Gästen kann detailliert verfolgt werden

Zusammenfassung

Das Heranziehen des Analytic-Tools *Google Analytics* zeigt, dass in diesem Themenbereich Facebook Insights allein nicht alle Bereiche abdeckt. Hierfür gibt es eine Reihe etablierter Tools mit Spezialisierung auf unterschiedlichste Fälle. Und so sollte das Thema auch gehandhabt werden: Je nach Anwendung ändern sich Anforderungen und damit das am besten passende Tracking-Tool.

Ein konkretes Beispiel hierfür ist *Parse Analytics*. Verwendest du in deiner Anwendung die Nutzerregistrierung mit Parse, wirst du in *Parse Analytics* weitere interessante Informationen finden. In Abbildung 10-11 zeigen wir die Metrik »Retention«, die über die Rückkehrquote zur Anwendung aufklärt. In unserem (fiktiven) Testbeispiel kam lediglich ein Nutzer nach der Registrierung am nächsten Tag zur Anwendung zurück.

Abbildung 10-11: Die Rückkehrquote registrierter Nutzer in Parse Analytics

Auch wenn dieses Kapitel am Ende des Buches eingeordnet ist, sollte das Thema »Analytics« nicht an so später Stelle behandelt werden. Es empfiehlt sich, bereits in der frühen Phase der Anwendungsentwicklung das Thema einzuplanen, um entsprechende Testdaten zu erhalten.

Über die Autoren

Die Visitenkarte von **Stephan Alber** trägt in großen Buchstaben die Inschrift "IS A DEVELOPER". Hinter der kurzen Berufsbezeichnung liegen über 15 Jahre Erfahrung als Web-Entwickler. Im Alter von 18 Jahren gründete er sein erstes Internet-Startup, nach dem Abschluss des Studiengang Online Medien an der Hochschule Furtwangen führte sein Weg in die Welt der Werbeagenturen. Hierbei war er für namhafte Kunden wie Mercedes-Benz, Hugo Boss, L'Oréal Paris und Maybelline New York mit einer Spezialisierung auf Facebook-Anwendungen tätig. Als Nebentätigkeit ist er Autor für Facebook Developer Themen bei Allfacebook.de, darüber hinaus unterrichtet er über Facebook-Anwendungsentwicklung als Konferenz-Speaker und Seminarleiter. Für diesen Themenbereich wagte er im Jahr 2012 als Ausbilder den Sprung nach New York City (USA). Dort ist er bis heute für die technische Umsetzung von Marketing-Kampagnen für Kunden wie Intel, Unilever, J.P. Morgan Chase & Co und Nespresso verantwortlich.

Klaus Breyer ist Gründer und Geschäftsführer der buddybrand GmbH, dort verantwortlich für Technologien und Innovationen. Als Social-Media-Agentur begleitet buddybrand internationale Markenunternehmen wie KODA International, airberlin, Heineken oder Storck (merci, Toffifee) durch die digitale Transformation, entwickelt Kampagnen und ist für Content- und Commmunity-Management zuständig. Immer an spannenden digitalen Geschäftsmodellen interessiert ist er in der deutschen Start-Up- und Technologie-Szene gut vernetzt. Seine Stärke ist es, technische Inhalte einfach und anschaulich zu erklären. Er ist deshalb gern gesehener Speaker auf Konferenzen und aktiver Blogger auf klaus-breyer.de

Kornelius Nägele ist freier Webentwickler und Innovationsmaster. Er hat nach seinem Studium der Informatik an der Hochschule Konstanz ein Masterstudium an der Hochschule Esslingen in Innovationsmanagement abgeschlossen. Begonnen hat er seinen beruflichen Werdegang mit der Erstellung von Homepages und Onlineshops. Der Schwerpunkt seiner heutigen Arbeit liegt zum einen in der Entwicklung von Webanwendungen und zum anderen in der Optimierung von Innovationsprozessen für Großunternehmen.

Kolophon

Das Tier auf dem Einband von Praxishandbuch Facebook-Programmierung ist der Mornellregenpfeifer (*Charadrius morinellus*), meist nur Mornell genannt, und ist ein Vertreter der Eigentlichen Regenpfeifer (*Charadriinae*). Der Vogel brütet in den Tundren Eurasiens am und nördlich des Polarkreises. Vom Mornellregenpfeifer sind keine Unterarten bekannt. In der wissenschaftlichen Literatur wird häufig auch die Gattungsbezeichnung *Eudromias* verwendet.

Der Mornell ist zwar etwas kleiner als eine Amsel, wirkt jedoch aufgrund seiner langen, gelben Beine deutlich größer. Wichtigstes Feldkennzeichen im bunten Prachtkleid ist der breite, bis in den Nacken verlaufende weiße Überaugenstreif. Die Gesamtlänge von

Schnabelspitze zur Schwanzspitze beträgt im Durchschnitt 21 Zentimeter. Zwischen den Geschlechtern bestehen keine Größenunterschiede.

Der Mornell ist in seinem gesamten Verbreitungsgebiet ein obligater Zugvogel, mit einem relativ kleinen Überwinterungsgebiet im nördlichen Afrika sowie im Nahen Osten. Von Ende Juli an werden die Brutplätze geräumt, wobei die Weibchen etwa drei Wochen vor den Männchen und Jungvögeln abziehen.

Der schwedische Naturfotograf und -schriftsteller Bengt Berg widmete dem Mornellregenpfeifer das Buch *Mein Freund, der Regenpfeifer* (1925), in dem er seine Erlebnisse im schwedischen Fjäll beschreibt.

Die Umschlagabbildung stammt aus *Johnson's Natural History*. Die Schriftart auf dem Einband ist Adobe ITC Garamond. Die Schriftart für den Text ist Linotype Birka. Die Schrift für die Überschriften heißt Adobe Myriad Condensed, und als Schriftart für den Code haben wir TheSans Mono Condensed von LucasFont verwendet.