O'REILLY®

Praxishandbuch Facebook-Programmierung

FACEBOOK-ANWENDUNGEN MIT JAVASCRIPT UND PHP

Stephan Alber,
Klaus Breyer, Kornelius Nägele

Inhalt

KAPITEL 4

Open Graph und soziale Plugins

Ein elementarer Bestandteil eines jeden sozialen Netzwerks ist das Teilen von Links. Die technische Grundlage für verwandte Funktionen innerhalb der Facebook-Plattform sind das Open-Graph-Protokoll und die sozialen Plugins (*engl. Social Plugins*).

Das Open-Graph-Protokoll ist ein offener Web-Standard. Mit dem Schema des Protokolls können Informationen über eine Webseite in Form von Meta-Tags definiert werden. Hierzu gehören unter anderem Titel, Beschreibung oder Bilder einer Webseite. Die Gesamtheit eines solchen Datensatzes wird als Open-Graph-Objekt bezeichnet.

Diese Meta-Tags dienen (sofern vorhanden) als Informationsquelle für jeden Link, der auf Facebook geteilt wird. Obwohl das Protokoll von Facebook entwickelt wurde, verwenden auch andere Anbieter wie Pinterest oder Google Plus diese Technologie als Ergänzung zu hauseigenen Standards.

Neben dem Posten von Links im Facebook-Newsfeed oder in der eigenen Timeline können Links mit Hilfe der sozialen Plugins direkt von der entsprechenden Webseite geteilt werden. Zu den sozialen Plugins gehören bekannte Facebook-Produkte wie der Like-Button oder die Kommentar-Box. Alle Plugins bedienen sich bei der Erstellung der Link-Vorschau des Open-Graph-Protokolls.

Auf den folgenden Seiten werden wir die Einrichtung der Meta-Tags sowie aller sozialen Plugins behandeln. Auf Custom-Open-Graph-Objekte werden wir im Kapitel 8, *Games,* eingehen.

Open-Graph-Protokoll

Bevor wir tiefer in das Thema *Open Graph* eintauchen, möchten wir klarstellen, dass allgemein zwei unterschiedliche Typen von Open-Graph-Objekten existieren: Das Open-Graph-Basisobjekt des Typs `website` sowie sogenannte Custom-Open-Graph-Objekte. Bei Custom-Open-Graph-Objekten handelt sich es um Objekte, die eine benutzerdefinierte Bezeichnung sowie Eigenschaften besitzen.

Wie bereits erwähnt werden wir zu einem späteren Zeiptunkt auf diese Objekt-Typ zurückkommen. Zunächst werden wir uns mit dem Open-Graph-Basisobjekt beschäftigen. Es beschreibt die grundlegenden Elemente einer Webseite:

og:title
: Der Titel der Seite. Hierbei handelt es sich nicht zwingend um den (allgemeinen) Namen einer Webseite, sondern um den Titel einer spezifischen Seite bzw. Unterseite. Ein Beispiel: Ein Artikel auf Spiegel online (*www.spiegel.de*) mit der Überschrift »Anonyme Anmeldung bei Apps: Zuckerberg verspricht Facebook-Nutzern mehr Kontrolle« wird selbige als Titel verwendet und nicht der Seitenname »Spiegel Online«. Für den Seitennamen sollte stattdessen das Property og:site_name verwendet werden. Dieses gehört allerdings nicht zu den Eigenschaften des Open-Graph-Basisobjekts. Für diesen Zweck sollte stattdessen das Property `og:site_name` verwendet werden, das allerdings nicht zu den Eigenschaften des Open-Graph-Basisobjekts gehört.

og:type
: Diese Eigenschaft beschreibt den Charakter der Seite. Der Standartwert ist `website`. Darüber hinaus können spezielle Typen wie `article` oder `video.movie` den Charakter genauer beschreiben. Abhängig vom Typ sind weitere Eigenschaften erforderlich. So wird der Typ `video.movie` die Länge des Films erwarten.

og:image
: Unverzichtbar: Das Vorschaubild zum Link einer Webseite. Alle Bildverweise müssen als absolute URLs angegeben werden. Relative oder absolute Pfade werden nicht aufgelöst.

og:url
: Wie auch bei Suchmaschinen kann das Problem entstehen, dass mehrere URLs zur selben Seite existieren. Um doppelte Einträge zu vermeiden, sollte für jedes Open-Graph-Objekt eine zentrale URL definiert werden. Diese URL ist auch unter dem Namen »Canonical URL« oder »kanonische URL« bekannt.

Die aufgelisteten Werte müssen in Form von `<meta>`-Tags im `<head>`-Bereich einer Seite platziert werden, d.h., die Tags ähneln anderen HTML-Meta-Tags wie dem `description`-Tag. Open-Graph-Tags bestehen aus einer Kombination von `property`- und `content`-Attributen. Im Attribut `property` wird der Name des Wertes festgelegt, in `content` der eigentliche Wert. Die Konfiguration der Open-Graph-Tags eines Basisobjekts könnte wie im folgenden Beispiel aussehen:

```
<head prefix="og: http://ogp.me/ns# fb: http://ogp.me/ns/fb#
    {dein-canvas-name}: http://ogp.me/ns/fb/{dein-canvas-name}#">
<meta property="og:type"
      content="website" />
<meta property="og:url"
      content="http://www.domain.com/index.html" />
<meta property="og:title"
      content="Dein Seitentitel" />
```

```
<meta property="og:image"
      content="https://www.domain.com/image.jpg" />
<meta property="fb:app_id"
      content="{deine-app-id}" />
```

Der Wert ist `fb:app_id` optional. Sofern er aktiviert ist, können im *Insights* Bereich der Anwendung Statistiken über das Objekt abgerufen werden.

Sollten keine Open-Graph-Meta-Tags vorhanden sein, wird der Facebook-Scraper eine Link-Vorschau auf Basis von anderen `<title>`- und `<meta>`-Tags oder Auszügen längerer Textpassagen des `<body>` erstellen.

Technischer Hintergrund: Der Facebook-Scraper

Jedes Mal, wenn ein (neuer) Link auf Facebook geteilt wird, löst dies eine HTTP-Anfrage der Facebook-Server auf die geteilte URL aus. In dieser Anfrage, die vom sogenannten *Facebook-Scraper* ausgeführt wird, werden alle Open-Graph-Meta-Tags ausgelesen und in der Facebook-Datenbank gespeichert. Der entsprechende Eintrag kann über die Graph-API-URL `https://graph.facebook.com/?id={URL}` ausgelesen werden. Für die URL *https://developers.facebook.com/docs/plugins/* würdet ihr etwa folgende Antwort erhalten:

```
{
   "og_object": {
     "id":  "10150101516232211",
     "description":  "Social plugins let you see what your
friends have liked, commented on or shared on sites across the web.",
     "title":  "Social Plugins",
     "type":  "article",
     "updated_time":  "2014-10-26T12:25:08+0000",
     "url":  "https://developers.facebook.com/docs/plugins/"
  },
   "share": {
     "comment_count": 69,
     "share_count": 1022265
  },
   "id":  "https://developers.facebook.com/docs/plugins/"
}
```

Der verwendete Pfad entspricht dem in der Tabelle 3-1 »Alle Graph-API-Endpunkte im Überblick « vorgestellten API-Endpunkt-`/url`. Wie zu sehen ist, werden nicht alle Open-Graph-Meta-Daten (wie `og:image`) über die API zurückgegeben.

Open Graph basierte Link-Vorschau

Ein Ziel des Protokolls ist wie erwähnt das Erzeugen einer Link-Vorschau. Das Testen deiner Link-Vorschau versteht sich von selbst: Du postest den jeweiligen Link auf Facebook und wählst die Privatsphäre-Einstellung »Nur ich«, wenn der Test nicht für deine Freunde sichtbar sein soll. Ein Beispiel für eine Link-Vorschau zeigen wir in Abbildung 4-1. Die Darstellungsgröße des Vorschaubildes beträgt momentan 470x246 Pixel bzw.

487x255 Pixel, je nach Kontext. Facebook empfiehlt, das Bild in einer Größe von mindestens 1200x630 Pixeln bereitzustellen.

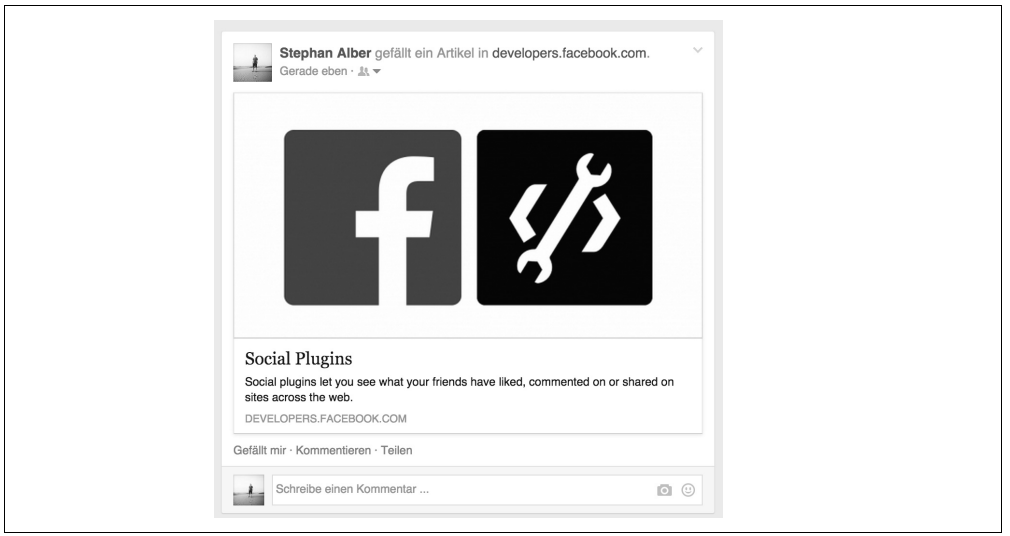

Abbildung 4-1: Link-Vorschau auf Basis des Open-Graph-Protokolls

Open-Graph-Object-Debugger

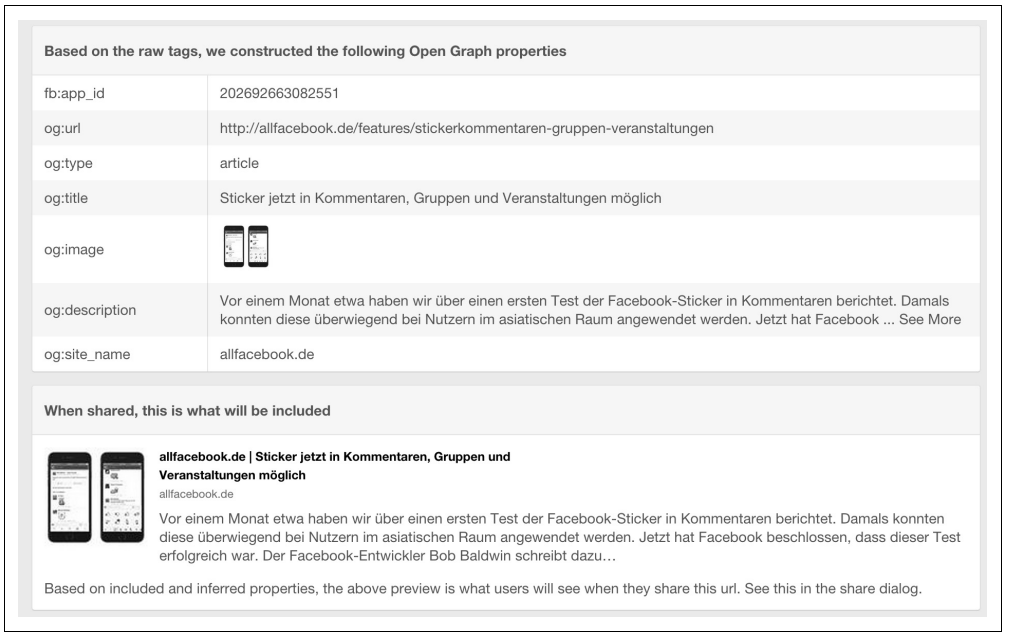

Abbildung 4-2: Überprüfung von Open-Graph-Meta-Tags durch den Open-Graph-Object-Debugger

Die Richtigkeit der Tags kann mit Hilfe des Open-Graph-Object-Debuggers überprüft werden. Der Debugger ist unter der URL *https://developers.facebook.com/tools/debug/* erreichbar. Eine beispielhafte Überprüfung ist in Abbildung 4-2 zu sehen. Der Bereich *»When shared, this is what will be included«* gibt Auskunft, welche Inhalte in der Link-Vorschau angezeigt werden. Sollte die Konfiguration ungültige Werte wie ungültige Bilderverweise enthalten oder gar wichtige Tags vermissen, klärt der Debugger über Fehler und Lösungsmöglichkeiten auf.

Abbildung 4-3: Beispiel für eine Fehler- bzw. Warnmeldung des Debuggers

Open-Graph-Cache erneuern

Die erfassten Daten werden für eine Stunde bis zu mehreren Tagen zwischengespeichert. Die Erneuerung des Caches kann über den Open-Graph-Object-Debugger (*https://developers.facebook.com/tools/debug/*) erzwungen werden. Sende eine Anfrage für die gewünschte URL, anschließend ergänzt sich das Formular um einen Button »Fetch new scrape information«. Betätige ihn, um das Laden neuer Daten initiieren.

Soziale Plugins

Das beschriebene Open-Graph-Protokoll ist ein elementarer Bestandteil der sozialen Plugins (*Social Plugins*). Als Social Plugins bezeichnet Facebook Produkte des JavaScript-SDK, die sich über sogenannte Snippets auf besonders einfache Weise und fast ohne tiefere Programmierkenntnisse in Webseiten einbinden lassen. Sie zielen in erster Linie auf das schnelle Teilen von Inhalten im Web, sind aber nicht darauf beschränkt. An dieser Stelle kommt das Open-Graph-Protokoll ins Spiel. Der bekannteste Vertreter der Social Plugins ist ohne Frage der *Like*-Button bzw. auf Deutsch der *Gefällt-mir*-Button.

Schnell-Übersicht

Zunächst eine Schnell-Übersicht der aktuell verfügbaren sozialen Plugins:

- Like-Button
- Share-Button
- Send-Button
- Embedded Posts
- Follow-Button

- Comments
- Activity Feed
- Recommendations Feed
- Like-Box
- Facepile

Der jeweilige Name lässt die Funktionalität des Plugins erahnen, wir werden auf die einzelnen Plugins noch im Detail eingehen.

Code-Generator

Für den Einbau von Social Plugins bietet Facebook per Plugin einen gesonderten Code-Generator. Über die Generatoren lassen sich über ein Formular sämtliche Plugin-Optionen anpassen, wie in Abbildung 4-4 am Beispiel des Like-Button-Plugins zu sehen. Ein Klick auf *Get Code* liefert ein HTML-Code-Snippet mit individuellen Konfigurationswerten. Es ist darauf zu achten, dass der Generator auch das Laden des JavaScript-SDK mitliefert. Sofern er im jeweiligen Projekt bereits geladen wird, darf dieser Teil selbstverständlich nicht übernommen werden.

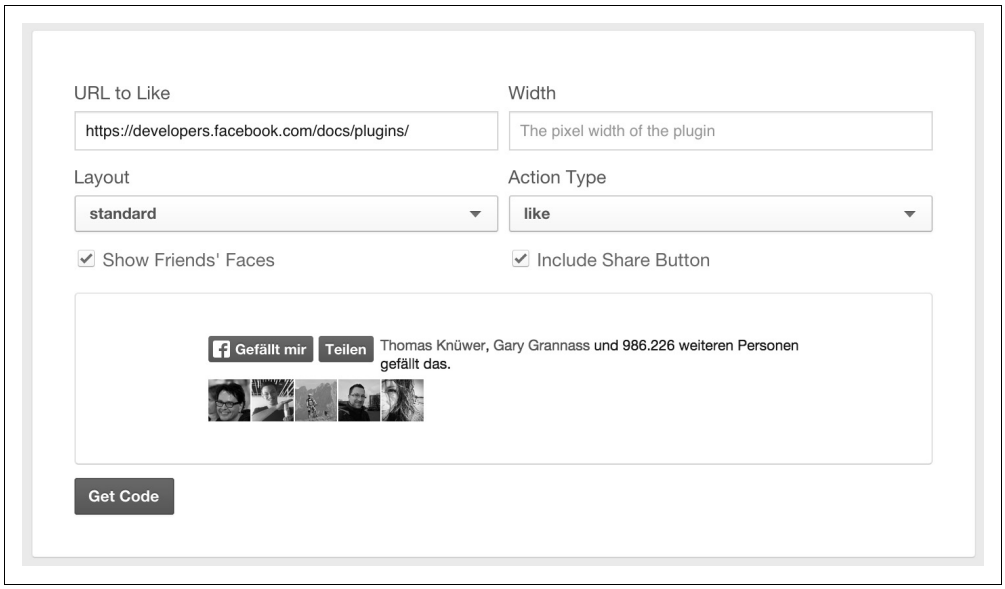

Abbildung 4-4: Social-Plugin-Snippet-Generator

HTML5, XFBML, IFrame oder URL

Für die meisten Plugins gibt es vier unterschiedliche Wege, sie einzubauen: via HTML5, XFBML, IFrame oder händisch per URL.

HTML5

Die gängige und empfohlene Technologie ist HTML5. Dies erfordert das Laden des JavaScript-SDK, das beim Laden einer Seite die HTML-Tags in die eigentlichen Social Plugins umwandelt. Jeder Tag zeichnet sich durch eine CSS-Klasse mit dem Prefix `fb-` sowie dem Namen des Plugins aus. Über data-Attribute werden Optionen festgelegt:

```
<div class="fb-like" data-href="..." data-layout="standard"
    data-show-faces="true"></div>
```

XFBML

XFBML steht für eXtended FaceBook Markup Language und ist eine Auszeichnungssprache, vergleichbar mit HTML. Die Tags werden wie auch in der HTML5-Variante über das JavaScript-SDK in Social Plugins umgewandelt. Alle Tags sind mit einem Suffix `fb` benannt, Optionen werden über die Attribute des Tags festgelegt:

```
<fb:like href="..." layout="standard" show_faces="true"></fb:like>
```

Die Verwendung von XFBML ist nicht zu empfehlen. Der Standard stammt aus den Anfangszeiten der Facebook-Plattform. Seinerzeit wurden Canvas-Anwendungen von Facebook ausgeliefert und dabei die XFBML-Tags serverseitig bereits in HTML umgewandelt. Hierzu gehörten Tags wie `<fb:name>` zur Anzeige eines Nutzernamens oder `<fb:profile-pic>` für die Darstellung eines Profilbildes. Diese Tags werden zum größten Teil noch unterstützt, werden früher oder später jedoch vermutlich vollständig von der Facebook-Plattform verbannt.

IFrame

Ob HTML5 oder XFBML: Beide Methoden generieren letztendlich Iframes, in denen die Plugins geladen und dargestellt werden. Daher ist in bestimmten Fällen die Verwendung dieser Methode direkt zu empfehlen. Dies ist unter anderem der Fall, wenn der Facebook-JavaScript-SDK nicht verwendet wird. Die Implementierung via Iframe kann deshalb die Ladezeiten beschleunigen. Nachteile sind allerdings mögliche Darstellungsfehler sowie ein unübersichtlicher Quellcode:

```
<iframe src="//www.facebook.com/plugins/like.php?href=...&layout=standard&show_
faces=true"></iframe>
```

Einzelne Plugins, wie die Recommendations Bar, bieten keine Implementierung über einen Iframe an, da das JavaScript-SDK zur Positionierung des Plugins benötigt wird.

URL

Für alternative Implementierungen können Plugins auch über eine HTTP-URL angesprochen werden. Die generierte Mini-Webseite kann u. a. in Web-Views von iPhone oder Android Apps verwendet werden:

```
https://www.facebook.com/plugins/like.php?href=...&layout=standard&show_faces=true
```

Social Plugins im Detail

Im Folgenden werden wir alle Plugins sowie ihre Einbindung mittels HTML5 vorstellen. Bekanntermaßen muss das JavaScript-SDK für die Verwendung initialisiert werden (siehe *Implementierung des JavaScript-SDK*). Achte unbedingt darauf, dass die Option `xfbml` auf `true` gestellt ist.

Alle Plugins, die auf eine Webseite verweisen, verwenden Open-Graph-Meta-Tags für die Erstellung einer Link-Vorschau. Eine Individualisierung der Vorschauelemente (Bild, Titel und Kurzbeschreibung) direkt im Plugin ist daher nicht möglich. Die Open-Graph-URL wird einheitlich über das Konfigurationsattribut `data-href` festgelegt.

Like-Button

Der Like-Button (*Gefällt-mir-Button*) ist ein schneller und einfacher Weg, um Inhalte zu teilen. Mit einem einzigen Klick kann ein Nutzer zeigen, dass ihm bzw. ihr Webinhalte gefallen. Ebenso kann der *Like*-Button mit einer Facebook-Fanseite über deren URL verbunden werden. Der entsprechende Inhalt wird durch den Klick gleichzeitig auf Facebook geteilt.

Im Folgenden ist ein Code-Beispiel für einen *Gefällt-mir*-Button zu sehen. Über die Klasse `fb-like` wird der Typ des Social Plugins definiert, weitere Argumente wie `data-href` individualisieren den Button für den gewünschten Verwendungszweck:

```
<div class="fb-like" data-href="https://developers.facebook.com/" data-layout="standard"
data-action="like" data-show-faces="true" data-share="true"></div>
```

Als Ergebnis erhalten wir einen *Gefällt-mir*-Button für die URL *https://developers.facebook.com/* mit einem Auszug von Freunden, denen diese URL ebenfalls gefällt:

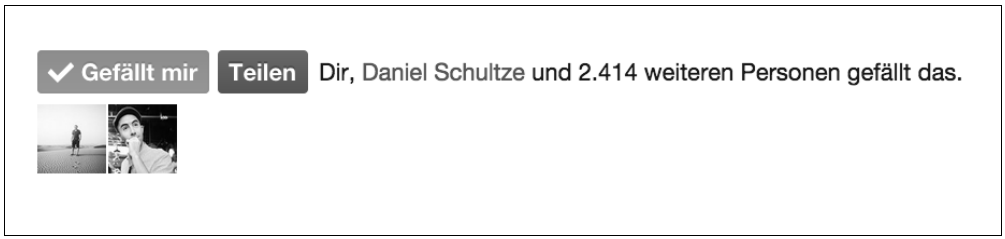

Abbildung 4-5: Darstellungsbeispiel eines Gefällt-mir-Buttons

Der *Gefällt-mir*-Button ist eine einfache und schnelle Möglichkeit für Nutzer, den eigenen Freunden auf Facebook Webinhalte mitzuteilen. Ein einfacher Klick genügt, um Apps, Webseiten oder Facebook-Seiten auf Facebook zu teilen. Es bietet sich an, direkt neben dem *Gefällt-mir*-Button auch einen Teilen-Button einzusetzen.

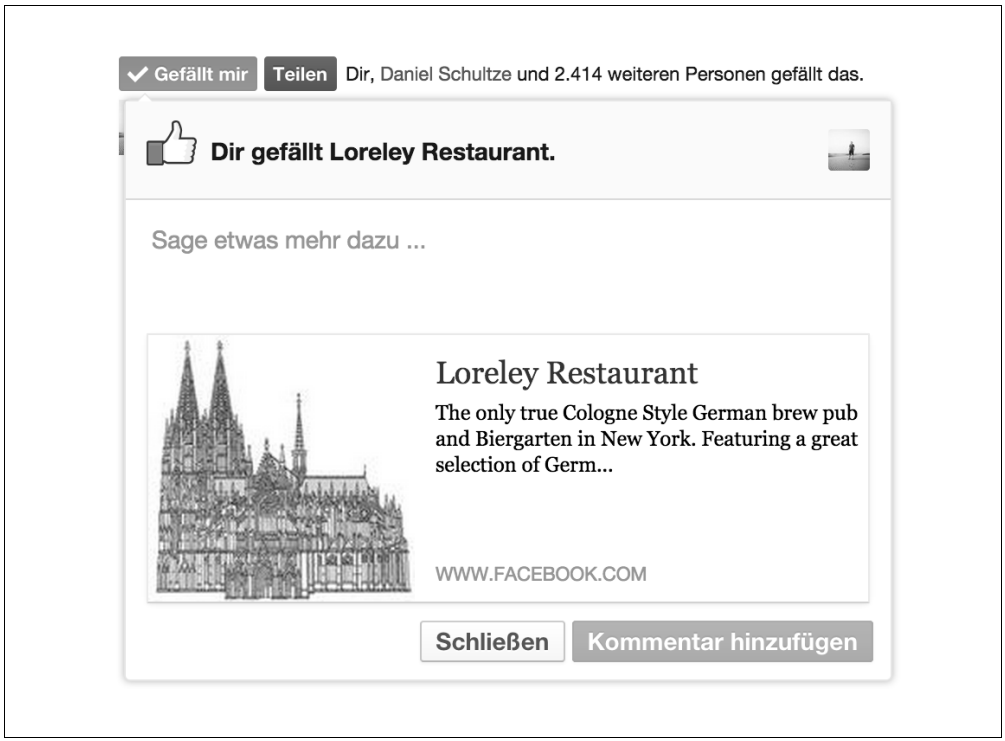

Abbildung 4-6: Darstellungsbeispiel eines Gefällt-mir-Buttons

Share-Button

Der Share-Button (*dt. Teilen-Button*) ermöglicht Nutzern primär, Webinhalte in der eigenen Chronik zu veröffentlichen. Der Share-Button (dt. Teilen-Button) ermöglicht Nutzern Webinhalte in der eigenen Chronik zu teilen. Darüber hinaus können aber auch andere Ziele für das Teilen ausgewählt werden: die Chronik eines Freundes, eine Gruppe oder in einer Facebook-Seite, die vom Nutzer verwaltet wird. Alternativ kann ein Inhalt über eine eine private Nachricht geteilt werden. Diese Variante entspricht der Funktionalität des Send-Button.

Mit der Möglichkeit, einen kurzen Kommentar dazu zu schreiben, ist der Teilen-Button ein wertvolles Werkzeug für die Verbreitung von Apps und App-Inhalten.

Der Button kann über folgendes HTML-Snippet eingebaut werden:

```
<div class="fb-share-button"
    data-href="http://www.oreilly.de/"
    data-width="100"></div>
```

Die Darstellungsweise des Buttons sowie des Share-Dialogs sind in Abbildung 4-7 zu sehen.

Abbildung 4-7: Facebook-Teilen-Dialog

Größe und Design des Standard-Buttons sind etwas minimalistisch gehalten. Erfreulicherweise kann derselbe Dialog auch über den Aufruf der JavaScript-Methode `FB.ui()` erzeugt werden, was den Einsatz eines benutzerdefinierten Buttons ermöglicht bzw. erleichtert. Eine solche händische Implementierung kann wie folgt aussehen:

```
<script>
function share() {
  FB.ui({
      method: 'share',
      href: 'http://www.oreilly.de/',
    },
    function(response) {
      if (response && !response.error_code) {
        alert('Posting completed.');
      } else {
        alert('Error while posting.');
      }
    }
  );
}
</script>
<a href="#" onclick="share(); return false">Teilen</a>
```

Embedded Posts

Um Facebook-Inhalte auf Webseiten oder Blogs zu teilen, gibt es das Plugin Embedded Posts. Es erlaubt, öffentliche Posts von Facebook als HTML-Code »mitzunehmen« und

direkt im Quellcode einzubetten. Der entsprechende Facebook-Post wird über die URL des Posts im Attribut `data-href` festgelegt:

```
<div class="fb-post" data-href="https://www.facebook.com/FacebookDevelopers/posts/
10152103052743553" data-width="500"></div>
```

Abbildung 4-8: Beispiel für einen »Embedded Post«

Ein Darstellungsbeispiel für einen Embedded Post ist in Abbildung 4-8 zu sehen. Das Design ist von Facebook vorgegeben und kann, abgesehen von der Breite (mittels data-width), nicht angepasst werden. Angemeldete Nutzer können nicht direkt mit dem Post interagieren, allerdings die Seite des Posts »liken«.

Comments

Mit dem Kommentar-Plugin ist es möglich, auf einer Webseite oder in einer App eine Kommentarfunktion einzubinden. In Facebook angemeldete Nutzer können so auf die Eingabe von Name, E-Mail-Adresse oder eines Captcha-Codes verzichten. Das Plugin regelt im Hintergrund die Authentifizierung und dient gleichzeitig der Spam-Abwehr.

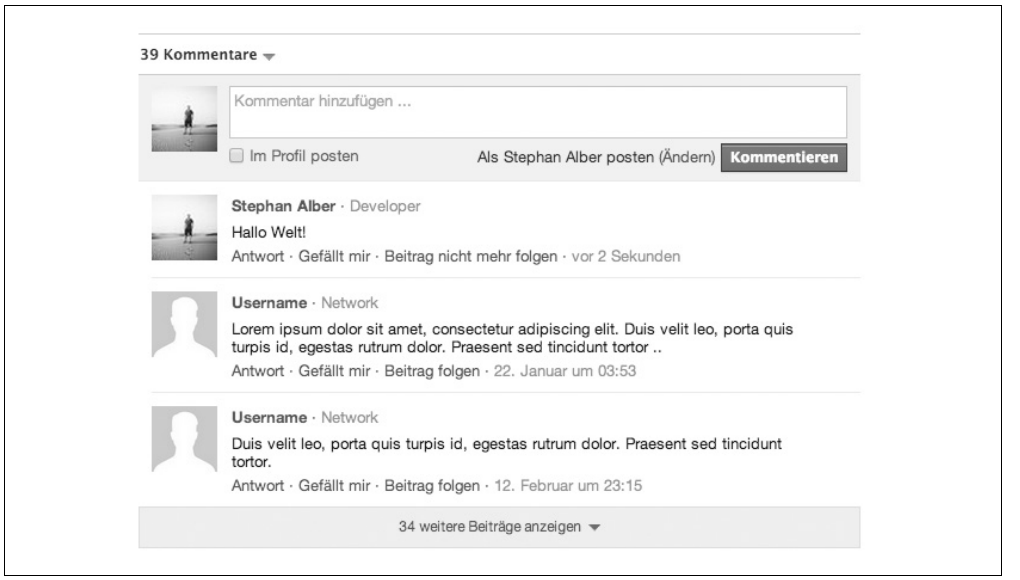

Abbildung 4-9: Beispiel für eine Kommentar-Box.

Wie im Screenshot zu sehen, ist es möglich, einen Kommentar auch im Profil bzw. der eigenen Timeline zu veröffentlichen. Das Ergebnis der Veröffentlichung zeigen wir in Abbildung 4-10.

Abbildung 4-10: Veröffentlichung eines Kommentars im Profil

Ein Kommentar-Strang bezieht sich jeweils auf die Seite, mit deren eindeutiger URL das Plugin identifiziert wurde. D.h., prinzipiell ist es möglich, dieselben Kommentare auf unterschiedlichen Seiten anzuzeigen, Klicks auf einen Kommentar, der im Profil veröffentlicht wurde, führen hierdurch verständlicherweise gegebenenfalls auf eine falsche Zielseite.

Die URL wird wie auch bei anderen Plugins über das Attribut `data-href` gesteuert, dazu via `data-numposts` die Anzahl der angezeigten Kommentare. Das HTML-Snippet für das Comments-Plugin gestaltet sich wie folgt:

```
<div class="fb-comments" data-href="http://developers.facebook.com/docs/plugins/comments/
" data-numposts="5" data-colorscheme="light"></div>
```

Send-Button

Der Send-Button ermöglicht Nutzern das private Teilen von Webinhalten mit Freunden oder Gruppen. Das HTML-Snippet für den Send-Button gestaltet sich wie folgt:

```
<div class="fb-send" data-href="https://developers.facebook.com/docs/plugins/"></div>
```

Der Button kann als Plugin optisch nicht angepasst werden. Verwende für diesen Zweck einen eigenen Button und die entsprechende Dialog-Funktion im JavaScript-SDK:

```
FB.ui({
  method: 'send',
  link: 'http://www.domain.com',
});
```

Den Send-Button im geöffneten Zustand sowie das Ergebnis eines Send-Vorgangs zeigen wir in den beiden folgenden Abbildungen. Tipp: Zum Testen der Funktion kannst du auch eine Nachricht an dich selbst senden.

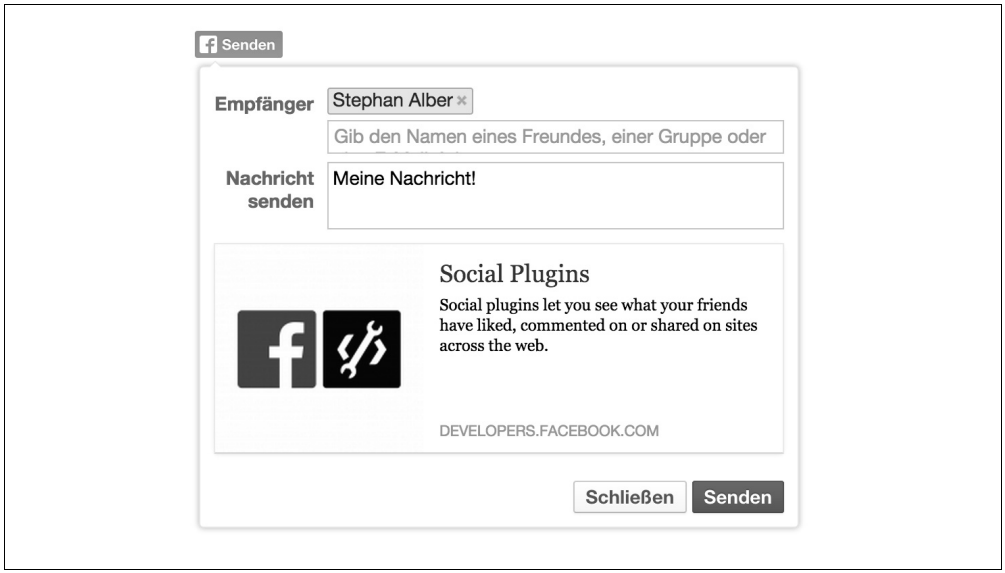

Abbildung 4-11: Send-Button

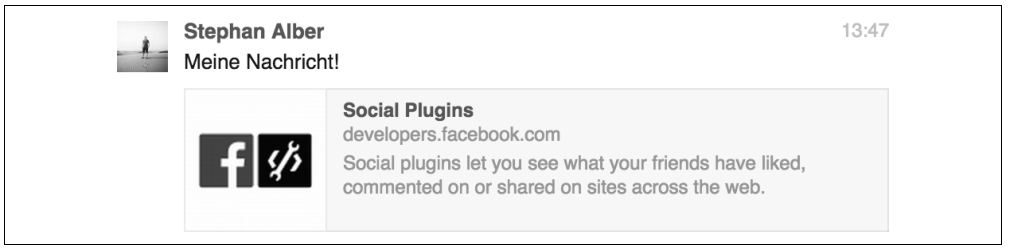

Abbildung 4-12: Eine durch den Send-Button generierte Nachricht

Follow-Button

Nutzer können mit dem Abonnieren-Button die öffentlichen Statusmeldungen von Facebook-Nutzern verfolgen. Facebook versucht allerdings, die unwichtigen Updates herauszufiltern, wenn der Nutzer keine besonderen Einstellungen wählt. Diese Funktion ist für Blogger und andere Personen des öffentlichen Lebens von besonderem Interesse.

Als Quelle für das `data-href`-Attribut dient in diesem Fall die URL zum Facebook-Profil der betreffenden Person, z.B. *https://www.facebook.com/zuck* für Mark Zuckerbergs Facebook-Profil. Verwende folgendes HTML-Snippet, um einen solchen Follow-Button in eine Webseite einzubinden:

```
<div class="fb-follow" data-href="https://www.facebook.com/zuck" data-width="500" data-
height="100" data-colorscheme="light" data-layout="standard" data-show-faces="true"></div>
```

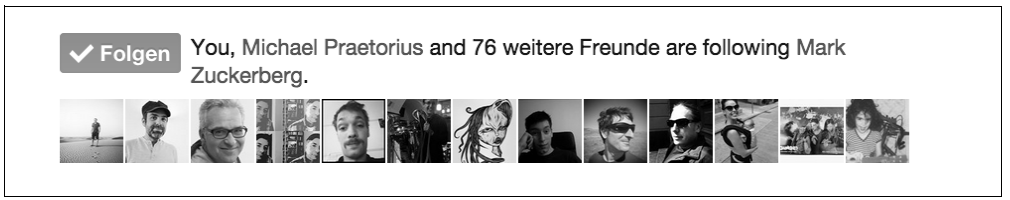

Abbildung 4-13: Der Follow-Button am Beispiel von Mark Zuckerbergs Profil

Activity Feed

Mit dem Activity Feed lassen sich Statusmeldungen, die in Verbindung mit einer Webseite erzeugt wurden, in die Webseite einbinden. Im Feed werden nur öffentliche Statusmeldungen oder Statusmeldungen von Freunden angezeigt. Damit könnte eine Meldung beispielsweise das Empfehlen eines Artikels durch einen Freund sein.

In seiner Dokumentation bietet Facebook zwar Optionen für die Anwendungs-ID (`data-app-id`) und die Domain (`data-site`), über die die Meldungen erzeugt werden, allerdings werden sie über die Einstellung im JavaScript-SDK sowie die aktuell verwendete Domain automatisch festgelegt bzw. erzwungen.

Das HTML-Snippet für das Activity Feed gestaltet sich wie folgt:

```
<div class="fb-activity" data-action="likes, recommends" data-height="400" data-
colorscheme="light" data-header="true"></div>
```

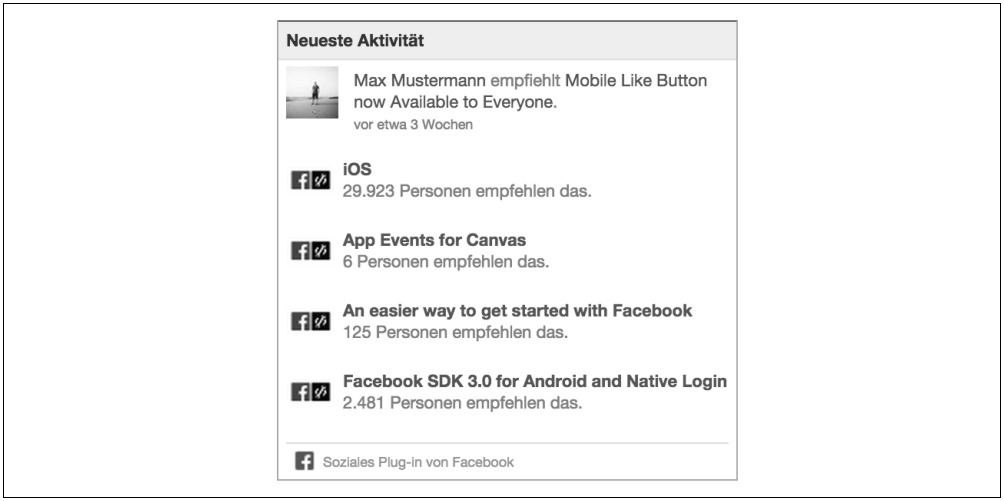

Abbildung 4-14: Beispielansicht für eine Activity Feed Box

Recommendations Feed

Das Recommendations Feed ist ein Plugin insbesondere für Betreiber von Seiten mit redaktionellen Inhalten oder Produkten. Es stellt die Inhalte der Webseite dar, die am häufigsten empfohlen wurden. Wie beim Activity Feed sind die Optionen für die Anwendungs-ID (`data-app-id`) und die Domain (`data-site`) kosmetischer Natur. Füge den Feed über das folgende HTML-Snippet zu deiner Webseite hinzu:

```
<div class="fb-recommendations" data-action="likes, recommends" data-width="300" data-
height="400" data-colorscheme="light" data-header="true"></div>
```

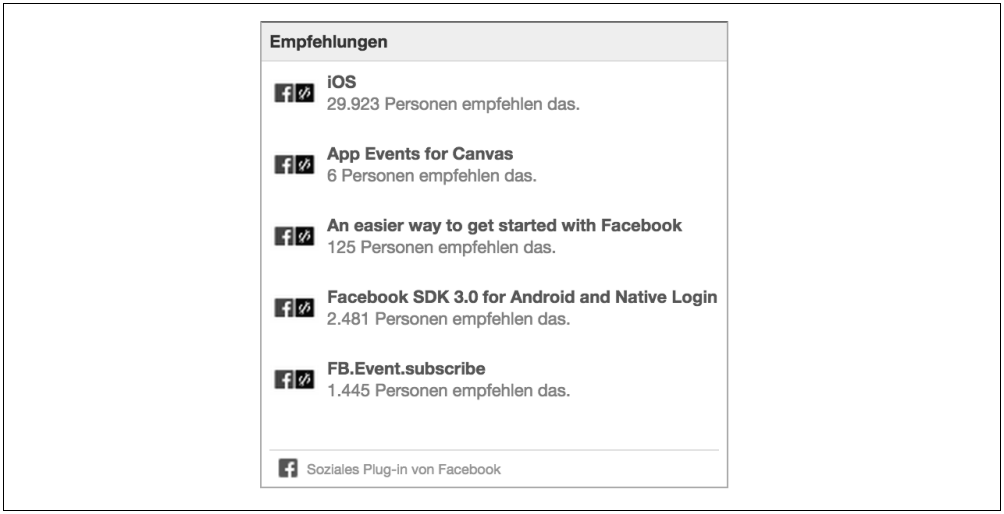

Abbildung 4-15: Beispielansicht für ein Recommendations Feed

Like-Box

Die Like-Box ist eine erweiterte Version des Like-Buttons, die allerdings nur für Facebook-Seiten funktioniert. Zusätzlich zur Like-Funktion wird ein Feed der neuesten Seitenbeiträge angezeigt. Verwende dieses HTML-Snippet, um eine Like-Box in eine Webseite einzubinden:

```
<div class="fb-like-box" data-href="https://www.facebook.com/FacebookDevelopers" data-
width="300" data-height="400" data-colorscheme="light" data-show-faces="true" data-
header="true" data-stream="true" data-show-border="true"></div>
```

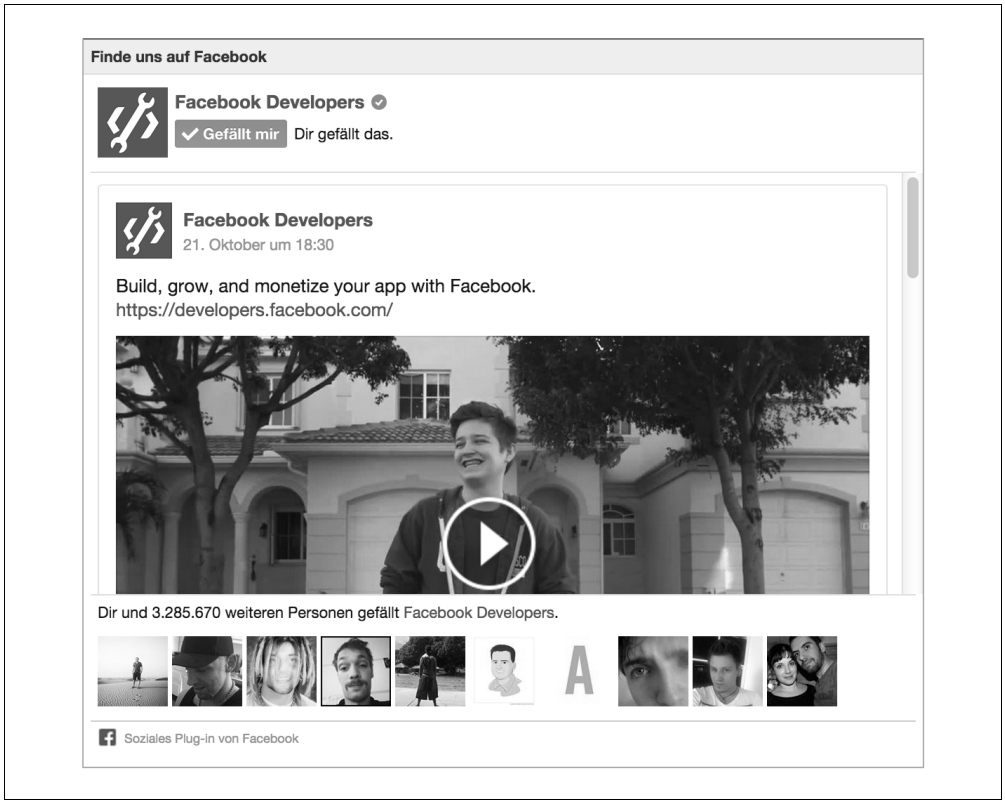

Abbildung 4-16: Beispiel für eine Like-Box mit Seitenbeiträgen

Facepile

Über das Facepile-Plugin können Profilbilder von Freunden, denen eine bestimmte Facebook-Seite gefällt oder die eine bestimmte Facebook-Anwendung verwenden, dargestellt werden, ähnlich wie beim Like-Button über das `show-faces=true`-Attribut. Neben den Profilbildern wird eine gewisse Anzahl Freunde, die zur entsprechenden Gruppe gehören, angezeigt. Profilbilder selbst werden nur angezeigt, wenn die Privatsphäre-Einstellungen des jeweiligen Nutzers dies zulassen, d.h., dass die Information über die jeweilige

Anwendungsnutzung mit Freunden geteilt wird. Verwende dieses HTML-Snippet, um ein Facepile zu deiner Webseite oder Anwendung hinzuzufügen:

```
<div class="fb-facepile" data-href="https://www.facebook.com/FacebookDevelopers" data-
width="500" data-height="50" data-max-rows="1" data-colorscheme="light" data-size="small"
data-show-count="true"></div>
```

Abbildung 4-17: Facepile

Zusammenfassung

Im Kapitel 4 *Open Graph und soziale Plugins* haben wir das Zusammenspiel beider Parteien kennengelernt. In der Praxis können wir den Open Graph verwenden, um Seiten unserer Webseite oder Anwendung semantisch beschreiben.

Dies könnte beispielsweise das Profil eines Spielers bzw. Anwendungsnutzers sein. Teilt ein Facebook-Nutzer den Profil-Link über ein soziales Plugin oder manuell auf Facebook, werden die definierten Open-Graph-Meta-Attribute für die Link-Vorschau verwendet. Weitere Informationen zum Protokoll findest du auf der offiziellen Projekt-Webseite *http://ogp.me/*.

Insbesondere wurde uns bewusst, dass Open-Graph-Tags öffentlich zugänglich sein müssen, um sie zu testen. In bestimmten Projekten kann es dazu kommen, dass Testversionen vor der Veröffentlichung auf durch IP-Sperren gesicherten Systemen beheimatet sind. In diesem Fall sieht der Facebook-Scraper (oder auch Facebook-Crawler genannt) selbstverständlich schwarz, womit Funktionen wie das Teilen über den Share-Dialog zwar grundlegend funktionsfähig sind, die geteilten Beiträge allerdings weder Bild noch Titel oder Beschreibung anzeigen.

Um dieses Problem zu umgehen, müssen die IP-Adressen des Facebook-Crawlers zur Ausnahmeliste hinzugefügt werden. Nach aktuellem Stand sind dies folgende IP-Adressen:

```
31.13.24.0/21
31.13.64.0/18
66.220.144.0/20
69.63.176.0/20
69.171.224.0/19
74.119.76.0/22
103.4.96.0/22
173.252.64.0/18
204.15.20.0/22
2401:db00::/32
2620:0:1c00::/40
2a03:2880::/32
```

Weitere Informationen sowie die aktuellsten IP-Adressen findest du in der Facebook-Developer-Dokumentation unter *Sharing Best Practices for Webseites & Mobile Apps – 2. Facebook Crawler access* (*https://developers.facebook.com/docs/sharing/best-practices#crawl*).

Ein besonders angenehmer Nebeneffekt des Open-Graph-Protokolls ist die Erkenntnis, dass das Protokoll auch von anderen Betreibern ausgelesen wird. Daher: Optimieren wir unser Projekt für Facebooks Open-Graph-Protokoll, sind wir gleichsam bereits bestens für das Teilen auf Twitter, Google Plus und Pinterest vorbereitet.

Zudem haben wir alle sozialen Plugins im Detail erforscht. Innerhalb weniger Minuten können wir mit ihnen unsere Projekte um wichtige Funktionen erweitern. Alle Plugins funktionieren auch auf mobilen Endgeräten, sind

dafür allerdings meistens nicht optimiert.

Wie auch für die Open-Graph-Tags gilt beim Testen der sozialen Plugins die Regel, dass sie nur in einer öffentlich zugänglichen Umgebung vollständig prüfbar sind.

Nächste Schritte

In den ersten Kapiteln haben wir uns die Grundlagen der Facebook-Plattform angeeignet:

- Anwendungstypen und ihre Einrichtung
- Social Graph & Open Graph
- Facebook-Graph-API & SDKs

In den später folgenden Kapiteln *Facebook-Login* und *Anwendungsbeispiele* werden wir auf dieses Wissen erneut zugreifen. Doch zuvor werden wir als vorbereitende Übung im Kapitel 5 *Allgemeine Entwicklungstools* kleine Helfer wie JavaScript-Templates, Twitter-Bootstrap oder Facebooks Datenbank-Service Parse behandeln.