Kapitel 7
UI-Prinzipien und Konventionen

Alle Plattformen für mobile Endgeräte sind zwar recht ähnlich aufgebaut, aber in den Details zeigen sich dennoch die kleinen und großen Unterschiede. Neben UI-Prinzipien sind es vor allem Nutzungskonventionen, die diese Unterschiede ausmachen. Diese Konventionen wirken sich auf alle Apps aus, die für die jeweiligen Plattformen entwickelt werden.

Um als App-Designer für eine Plattform zielgruppengerecht gestalten zu können, ist es wichtig zu wissen, wie die einzelnen Betriebssysteme visualisiert und genutzt werden.

7.1 Jede Plattform ist anders

Ob iOS oder Android, beide Plattformen werden von Anwendern genutzt, um Aufgaben auszuführen, Spiele zu spielen oder beruflich damit zu arbeiten. Jedoch werden diese Aufgaben ganz unterschiedlich von den Plattformen realisiert.

Häufig sind es stillschweigende Nutzungskonventionen, die sich etabliert haben, die aber auf keiner dokumentarischen Basis fußen. Die eine Android-App lässt sich von einer anderen inspirieren, bestimmte Bestandteile werden übernommen und tradiert. Das eine Erfolgsrezept orientiert sich an einem anderen – und das ist auch gut so. Denn auch Anwender übertragen gemachte Erlebnisse auf neue Apps und verlassen sich darauf, dass etwa die Anordnungen von Icons oder Bedienelementen ähnlich sein werden.

7.1.1 Die Erwartungshaltung des Nutzers

Nahezu 100 % der Smartphone-Nutzer verwenden Android- oder iOS-Geräte. Viele App-Designer konzentrieren daher ihre Bemühungen auf diese beiden großen Plattformen. In dem Fall, dass Sie für beide Plattformen eine App entwickeln möchten, sollten Sie die Eigenheiten und die Erwartungshaltung beider Nutzergruppen kennen und diese bei der App-Gestaltung berücksichtigen.

Die Erwartungshaltung des Nutzers können wir nur dann kennen, wenn wir verstehen, wie sich diese Plattformen bei der Nutzung auf den Nutzer auswirken. Daher wollen wir im Folgenden zunächst betrachten, wie bestimmte Aspekte auf den ein-

zelnen Betriebssystemen realisiert sind. Fünf Aspekte sollen miteinander verglichen werden:

- Gestaltung
- Cognitive Load
- Effizienz und Integration
- Individualisierbarkeit
- User Experience Friction

Gestaltung

In der Gestaltung bestehen bestimmte Konventionen, die wiederholt aufgegriffen und deshalb vom Benutzer vermutet bzw. vorausgesetzt werden. Jede Plattform hat dementsprechend ihre ganz eigene typische Optik. Gestalten Sie plattformspezifisch, erfüllen Sie die optische und ästhetische Erwartungshaltung der Plattformen. Bis vor wenigen Jahren gab es gestalterisch große Unterschiede zwischen den Plattformen. In den letzten Jahren bieten alle Plattformen im Zuge des Flatdesigns eine ähnliche gestalterische Basis, zumindest wirken viele Apps ähnlich in der Ausführung bzw. halten sich an stille Konventionen.

Mit iOS 7 wurde eine große Stiländerung auf der iOS-Plattform umgesetzt. Vom realistischen bzw. skeuomorphischen Gestaltungsprinzip wurde abgelassen, bei dreidimensionalen Elementen wurde ab Version iOS 7 nun komplett auf Ornamentik verzichtet (Abbildung 7.1). Das Design ist minimalistischer und orientiert sich an den Grundsätzen des Flatdesigns. Dies gilt auch für das aktuelle iOS 12.

> **Skeuomorphismus**
>
> Skeuomorphismus kommt aus dem Altgriechischen und bedeutet in etwa Behälter, Werkzeug oder Gestalt. Skeuomorphismus ist eine Design-Stilrichtung, bei der Objekte in ihrer Gestaltung die Form eines vertrauten Objekts nachahmen.

Abbildung 7.1 Links ist iOS 9 zu sehen, rechts Android 6.0 Marshmallow.

Es hat einen Grund, warum Apple die UI-Gestaltung nur selten stark verändert – große Veränderungen irritieren den Anwender und fordern von ihm Zeit ein, sich an die neue Gestaltung zu gewöhnen. Besonders wenn man bereits seit Längerem iOS-Nutzer ist, wirkt ein neues Design zunächst sehr fremd und gewöhnungsbedürftig.

In Android mit Marshmallow 6.0 haben die Grundzüge des minimalistischen Flatdesigns bereits etwas früher Einzug gehalten (Abbildung 7.1). Dabei geht Android einen Schritt weiter und erlaubt beim sogenannten Material Design die Nutzung von Ebenen und daraus resultierenden Schlagschatten. Für mich ist das Material-Design-Prinzip interessant, da es sich an der physischen Welt orientiert, ohne skeuomorphistisch zu sein.

Übrigens verfolgte zuerst Microsoft den Flatdesign-Ansatz, und zwar in Windows 10. In Form des viel gefeierten Metro-Designs hat sich dieser auch auf der Desktop-Plattform etabliert.

Abbildung 7.2 Windows war der Vorreiter in Sachen Flatdesign.

Cognitive Load

Der Cognitive Load ist die Summe von Elementen, die der Nutzer kennen muss, um ein Gerät spontan und intuitiv nutzen zu können. Es ist einer der Schlüsselaspekte der User Experience für wenig technisch versierte Nutzer. Er dient als Benchmark der User Experience für Plattformen und Applikationen. Eine von Pfeiffer Consulting

2013 durchgeführte Studie ergab die Ergebnisse aus Tabelle 7.1 zum Cognitive Load der jeweiligen Plattformen.

iOS 7	Android (Samsung)
Cognitive Load Index: 40	Cognitive Load Index: 162

Tabelle 7.1 Cognitive Load (je niedriger die Indexziffer, desto besser; Pfeiffer Report 2013)

Studie zum Cognitive Load

Die Ergebnisse der Studie von Pfeiffer Consulting können Sie unter *www.pfeiffer-report.com/v2/wp-content/uploads/2013/09/iOS7-User-Experience-Shootout.pdf* – allerdings nur auf Englisch – im Detail nachlesen. Momentan gibt es leider keinen Report, der die aktuellen Versionen der Betriebssysteme unter die Lupe nimmt. Jedoch haben sich die Plattformen erweitert, und die Ergebnisse spiegeln sich auch noch heute in den Betriebssystemen ein wenig wider.

Versuchen Sie also, den Nutzer unter Android nicht noch mehr zu verwirren, er hat schon genug mit dem Betriebssystem zu tun. Konzentrieren Sie sich bei Android besonders auf eine gelungene Usability und User Experience. Nutzen Sie beispielsweise die Vorzüge des neuen Material Designs, und versuchen Sie nur sehr wenige spezialisierte eigene Komponenten zu entwickeln.

Das bedeutet nicht, dass Sie diese Aspekte unter iOS etwas schleifen lassen können. Im Gegenteil, die Nutzer erwarten hier auch eine durchdachte und logische Nutzung des Betriebssystems.

Abbildung 7.3 Views bzw. Elemente, die ein iOS-7-Nutzer kennen muss

Im visuellen Vergleich zwischen iOS 7 und Android wird schnell erkennbar, warum Apple beim Cognitive Load so gut abgeschnitten hat. Auch wenn die Studie mittlerweile etwas betagt ist, lassen sich die Ergebnisse grundsätzlich auch auf iOS 12 und Android Pie übertragen.

Behalten Sie aber im Hinterkopf, dass die Betriebssysteme unterschiedliche Ansätze verfolgen, um einen Mehrwert für den Nutzer zu schaffen. Apple versucht bewusst, den Feature-Overload zu reduzieren. Das ist hingegen kein Anliegen von Android. Im

Gegenteil: Android sieht seine Stärke in der Vielzahl an Optionen, Apps, Widgets und Einstellungsmöglichkeiten, die es dem Anwender zur Verfügung stellt. Daran liegt auch das sehr schlechte Abschneiden beim Cognitive Load. Erfahrene Nutzer finden dies vielleicht sehr nützlich, unerfahrene Nutzer jedoch sind schlichtweg überfordert.

Abbildung 7.4 Views bzw. Elemente, die ein Android-(KitKat-)Nutzer unter Samsung kennen muss

Es sei aber noch angemerkt, dass Material Design nicht nur für mobile Applikationen entwickelt wurde, sondern auch für Desktop- bzw. Browseranwendungen. Ziehen Sie also nicht den Schluss, dass Material Design hinsichtlich des Cognitive Loads per se schlecht ist, sondern eher, dass die Anzahl der Komponenten alle möglichen Plattformen (Mobile Apps, Desktop-Apps, Web-Apps, Hybrid-Apps) abdecken kann. Nur bestimmte Komponenten sind relevant für das App-Design, nicht alle.

Eine weitere Anmerkung: Durch die Entwicklung von PWA- oder Hybrid-Apps können Sie Material Design auch für iOS-Apps nutzen! Hier erfahren Sie dazu mehr: *https://material.io/develop/ios/*.

Effizienz und Integration

Pfeiffer Consulting hat in der bereits oben genannten Usability-Studie den Betriebssystemen auch in puncto Effizienz und Integration auf den Zahn gefühlt. Bei diesem Punkt wurde die Integration wichtiger Funktionen wie Multitasking, Benachrichtigungen oder der Kamera getestet.

Hier stehen sich die zwei größten Plattformen in nichts nach (Tabelle 7.2). Sowohl Android als auch iOS schneiden mit 7 von 10 Punkten gut ab. Anwender können in beiden Betriebssystemen wichtige Funktionen einfach erreichen und mit diesen effizient arbeiten.

iOS 7	Android (Samsung)
Effizienzbewertung: 7/10	Effizienzbewertung: 7/10

Tabelle 7.2 Effizienz und Integration (je höher die Wertung, desto besser das Ergebnis; Pfeiffer Report 2013)

Effizienz hat etwas mit der Aufgabe zu tun, die man erledigen möchte. Je schneller diese gelöst wird, desto effizienter arbeitet das Betriebssystem. Mit der vom Betriebssystem geprägten Erwartungshaltung wird der Anwender auch an Ihre App herantreten.

Individualisierbarkeit

Ein weiterer wichtiger Faktor, der für die Usability und User Experience eines Geräts wichtig ist, ist seine Anpassungsfähigkeit an die individuellen Bedürfnisse seiner Nutzer.

Auf allen drei Betriebssystemen sind dem Nutzer Möglichkeiten gegeben, individuelle Anpassungen vorzunehmen. Daran hat sich auch in Android Pie oder iOS 12 nichts geändert. Der Nutzer kann mitunter ein Hintergrundbild wählen oder Schriftgrößen und Schriftschnitte individuell einstellen. Solche Einstellungen ermöglichen es dem Nutzer, sich stärker mit dem Gerät zu identifizieren. Andererseits können diese benutzerdefinierten Anpassungen auch negative Auswirkungen auf die Nutzung des Geräts haben. So kann eine filigrane Schrift für den Nutzer schlecht lesbar sein.

iOS 7	Android (Samsung)
Individualisierbarkeit: 6/10	Individualisierbarkeit: 7/10

Tabelle 7.3 Individualisierbarkeit (je höher die Wertung, desto besser das Ergebnis; Pfeiffer Report 2013)

In diesem Punkt schneidet Android am besten ab (Tabelle 7.3). Android war bereits von Beginn an als ein sehr individualisierbares Betriebssystem konzipiert, der Ursprung liegt im offenen Betriebssystem. Viele Apps bieten außerdem die Möglichkeit, tiefere Eingriffe in das System vornehmen zu können und diese nach individuellen Wünschen anzupassen. Apple hingegen setzt auf ein einheitliches und geschlossenes User-Interface-System, das nur wenig individualisierbar ist und ein einheitliches Look & Feel der Apple-Produkte garantieren soll. Dies hat einen positiven Effekt, da App-Anbieter sich an Apple orientieren können und so zielgerichtet Apps für die iOS-Nutzer entwickeln können. Diesen positiven Effekt gibt es unter Android nicht, das liegt in der Fragmentierung des Betriebssystems. Unter Samsung funktionieren bestimmte Funktionen anders als bei einem Huawei-Smartphone. Beide nutzen nicht die Version von Google, sondern ein Derivat davon.

Unter Android wird der Nutzer erwarten, dass er die App noch weiter anpassen und auf seine Bedürfnisse zuschneiden kann. Unter iOS ist diese Erwartung weniger gegeben, hier geht es restriktiver zu, und der Nutzer bewegt sich innerhalb strikterer Konventionen.

Geschlossenes vs. offenes System

Während Apple mit seinem Betriebssystem ein geschlossenes System anbietet, ist Android eine quelloffene Software. Für Nutzer ergeben sich daraus folgende Unterschiede: Beim Kauf eines Apple-Produkts können nur Dienste und Applikationen genutzt werden, die von Apple angeboten werden. Als Hauptbezugsquelle dient dabei Apples App Store. Anders bei Android: Hier können Hersteller (HTC, Samsung etc.) die Nutzeroberflächen ganz individuell anpassen und damit auch die Wünsche verschiedener Interessengruppen berücksichtigen. Auch können Apps aus unterschiedlichen App Stores bezogen werden.

User Experience Friction (UXF)

Unter dem Aspekt *User Experience Friction* (UXF) werden Stolperfallen bei der Nutzung beschrieben. Es geht um solche Momente, in denen das Gerät nicht das tut, was es soll, in denen es Schwachstellen in der Bedienung hat, oder den Fall, dass eine wichtige Funktion sogar ganz fehlt.

iOS 7	Android (Samsung)
UXF-Faktor: 17	UXF-Faktor: 30

Tabelle 7.4 UXF-Faktor (je niedriger der Wert, desto besser das Ergebnis; Pfeiffer Report 2013)

In der Usability-Studie von Pfeiffer Consulting schneidet das iOS-Betriebssystem bei diesem Punkt am besten ab (Tabelle 7.4). Bei Android sind es vorwiegend Inkonsistenzen innerhalb des Betriebssystems, die Verwirrung schaffen. Versuchen Sie, in Ihrer App derartige Stolperfallen zu vermeiden. Generell sollten Sie dafür sorgen, dass das Interface bzw. dessen gelernte Elemente das tut, was der User erwartet.

7.1.2 UI-Prinzipien der Plattformen

Jede der Plattformen wird mit Multitouch bedient, seit dem iPhone 6S/Plus und der Apple Watch haben diese Geräte zusätzlich einen drucksensitiven Bildschirm, unabhängig davon jedoch unterscheiden sich die Betriebssysteme in der eigentlichen Bedienung. Jedes der Betriebssysteme hat gängige Nutzungskonventionen, die ein Nutzer kennen und entsprechend in Apps anwenden muss.

Wenn Sie mit dem Gedanken spielen, Ihre App auf unterschiedlichen Betriebssystemen realisieren zu wollen, gibt es UI-Prinzipien, die Sie bei der Entwicklung und der Umsetzung Ihrer App-Idee berücksichtigen sollten. Sollten Sie also native Apps für die Plattformen entwickeln, kann das ein sehr umfangreiches Projekt werden, da Ihre

App auf jede Plattform zugeschnitten werden muss. Anders wäre es, wenn Sie eine Hybrid- bzw. PWA-Lösung entwickeln. Hier arbeiten Sie nur in einer Quellcode-Basis.

Obwohl die Grundidee und die Funktionalität Ihrer App sicherlich auf allen Plattformen abbildbar sind, sollten das Interface und die Handhabung der App im Idealfall auf die entsprechende Plattform zugeschnitten sein. Erfüllen Sie die Erwartungshaltung Ihres Nutzers, orientieren Sie sich an den etablierten Design Patterns und den nativen Komponenten, und respektieren Sie die nativen UI-Standards. Die wichtigsten stelle ich Ihnen im Folgenden vor.

Design Patterns

Design Patterns sind bewährte Schablonen. Diese Schablonen können App-Designer nutzen, um wiederkehrende Probleme zu lösen. So sind einige Abläufe, wie z. B. eine Registrierung, ein Stück weit standardisiert und können bei unterschiedlichen Projekten eingesetzt werden.

Zurück-Navigation

Die Zurück-Funktionalität ist eines der wichtigsten Navigationselemente für den Nutzer. Mit der Zurück-Funktion findet der Nutzer in tiefen hierarchischen Strukturen einer Anwendung stets den Weg zurück.

Unter iOS ist die Zurück-Option im oberen linken Bereich in der Navigationsbar angesiedelt, so wie man es von Webbrowsern her kennt. Der Zurück-Button ist ein reines Navigationsbedienelement. Er wird dazu genutzt, um zur vorherigen Seite innerhalb einer App zu gelangen. Die Option wird nicht dafür eingesetzt, um zwischen Applikationen zu wechseln. Seit iOS 9 gibt es zusätzlich einen App-Switcher, der es ermöglicht, zur vorherigen App zu springen. Dies funktioniert jedoch nur dann, wenn der Nutzer z. B. einen Internetlink in einer App wie Twitter berührt und sich daraufhin Safari öffnet. So kann der Nutzer von Safari zurück in Twitter springen. Diese Aktionsfläche befindet sich ganz oben links, wo sich die Anzeigen für Verbindungsstärke und Anbieter befinden (Abbildung 7.5).

Abbildung 7.5 Links die Zurück-Navigation von iOS, in der Mitte von Android und rechts die Aktionsfläche »Zurück zu Twitter«

Zurück unter iOS

Mit iOS 9 wurde ein neues Element eingeführt, das in der Statusbar angezeigt wird und das dann erscheint, wenn man von der einen App in die andere gelangt. So können manche Apps Verlinkungen ins Internet anzeigen. Bei einem Tap auf einen solchen Link wechselt der Nutzer in die Browser-App und kann dann durch den Link in der Statusleiste wieder zurück in die vorherige App gelangen.

Unter Android gibt es eine Up-Funktion und eine Zurück-Funktion. Die Up-Aktion befindet sich in der App-Bar und verhält sich wie die Zurück-Aktion unter iOS. Der Nutzer kann somit innerhalb einer App die Views in einer hierarchischen Struktur zurückgehen.

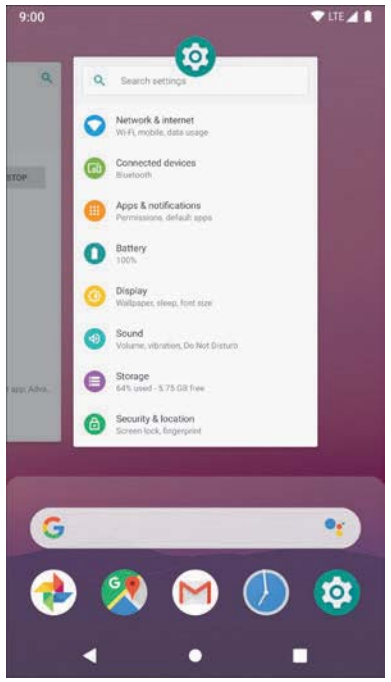

Abbildung 7.6 Multitasking-Ansicht unter Android

Im Gegensatz dazu verhält sich die Zurück-Option anders. Diese befindet sich bei älteren Geräten im unteren Bereich als Hardwareschaltfläche bzw. bei neueren Smartphones mit aktuelleren Betriebssystemen als Touchfläche. Dieser Zurück-Button wandert auch durch zuvor genutzte Hauptscreens zurück. So kann der Nutzer auch durch zuvor angezeigte Apps navigieren, sofern die aktuelle genutzte App in eine andere wechselt. Als Beispiel: Der Nutzer öffnet in der Twitter-App eine Verlinkung in das Internet, und der Browser öffnet sich. Mit dem Zurück-Button gelangt der Nutzer wieder zurück in die Twitter-App.

Abbildung 7.7 Up-Button bei Android

Unter Android wird ein Zurück-Button als Hardwareschaltfläche angeboten. Alternativ, wenn keine Hardwareschaltflächen vorhanden sind, kann durch eine Swipe-Geste vom unteren Bereich des Smartphones diese Schaltfläche eingeblendet werden. Auch hier wird chronologisch durch Apps oder besuchte Webseiten zurücknavigiert.

Tab-Navigation

Die Tab-Navigation ist ein Navigationselement, um zwischen Hauptbereichen einer App zu wechseln. Meist werden hier kleine Icons und Labels angezeigt. Bei Apps mit bis zu maximal fünf Seiten ist eine Tab-Navigation ein probates und sinnvolles Navigationsinstrument.

Unter iOS befindet sich die Tab-Navigation im unteren Bereich des Bildschirms. In der Tab-Navigation sollten nicht mehr als fünf Elemente angezeigt werden (warum mehr nicht empfehlenswert sind, erfahren Sie noch in Kapitel 8, »Von Controls, Views und Komponenten bis Atomic Design und Design-Systemen«). In manchen Apps wird hier auch die Toolbar angezeigt, die im Grunde so aussieht wie die Tabbar.

Abbildung 7.8 Tab-Navigation unter den Betriebssystemen (links iOS und rechts Android)

In Android-Apps wird empfohlen, die Tab-Navigation im oberen Bereich des Bildschirms zu positionieren. Dort wird Sie in der Regel von den Android-Nutzern erwartet. Unter Android dürfen mehrere Elemente angezeigt werden, und zwar in einem scrollbaren Container ❶. Der Nutzer muss nur seitwärts scrollen, wenn mehrere Elemente angezeigt werden.

Zusätzlich zur Tabbar unter Android gibt es neuerdings auch die sogenannte Bottom-Navigation. Diese ist im Grunde ähnlich zur iOS-Tabbar und sollte auch maximal fünf Elemente anzeigen.

Sollten Sie mehr als fünf Hauptbereiche in der App haben, können Sie als letzten Punkt den sogenannten More-Tab einfügen. Darunter befinden sich dann andere Bereiche der App. Meist verstecken sich dort sekundäre Funktionen, z. B. Support, Account, FAQ, oder Inhalte, die nicht wesentlich relevant für den Nutzer sind, z. B. Impressum und Rechtliches. Diese Inhalte werden dann in einem neuen View angezeigt.

Wechsel zwischen Ansichten

In iOS-Apps wechselt der Anwender zwischen Ansichten verschiedener Datensätze mittels der sogenannten *Segmented Control*. Ein Datensatz kann z. B. eine längere Liste sein. Jedes Segment innerhalb des Controls steht für eine weitere Ansicht mit Inhalt. So kann z. B. bei einer Shopping-App über die Segmented Control zwischen Beschreibung, Technische Details oder Rezensionen gewechselt werden.

Abbildung 7.9 Ansicht verschiedener Datenansichten

Bei Android-Apps wird der Spinner verwendet. Er funktioniert wie ein Dropdown-Menü und wird vorwiegend in der Action-Bar verwendet, die sich unter der App-Bar bzw. der Systembar befindet. Dieses Element kann auch als Navigationselement genutzt werden, während eine Tabbar genutzt wird, um innerhalb einer Inhaltsseite Ansichten zu wechseln.

Suchfunktion

Unter iOS befindet sich die Suche vorwiegend im oberen Bereich der App.

Abbildung 7.10 Suchfeld der Betriebssysteme

In Android-Systemen gibt es verschiedene Möglichkeiten, die Suche zu positionieren:

- als Search-Bar im oberen Bereich der App, ähnlich wie unter iOS; jedoch ist das Suchfeld hier häufig hinter einem Lupe-Icon versteckt, ein Tap darauf öffnet das Suchfeld.

- als Search-Widget; dieses Widget, das eigentlich keines ist, kann überall in der App eingesetzt werden, weshalb es als Widget bezeichnet wird; vorwiegend wird es in der Action-Bar im oberen Bereich positioniert.

Aktionen

Aktionen werden in Regel durch Icons oder Buttons repräsentiert, die bestimmte Funktionen darstellen. Sie erscheinen im Kontext einer View. Es sind dynamische Aktionen, die mitunter nur in bestimmten Screens angezeigt werden. Konkrete Beispiele wären: Speichern, Senden, Teilen, Liken etc.

In iOS-Apps können Aktionen an verschiedenen Stellen positioniert werden. So können Aktionsflächen als Leiste im unteren Bereich erscheinen oder in der Navigationsbar in der rechten oberen Ecke. Alternativ können diese Aktionsflächen natürlich auch im Zentrum des Bildschirms angezeigt werden, je nach Konzeption und Gestaltung der App.

Abbildung 7.11 Action-Bars

In Android-Apps wird empfohlen, die Aktionsflächen in der Action-Bar im oberen Bereich darzustellen. Sollten mehr Aktionen abgebildet werden, so wird ein Action-Overflow-Icon (auch gerne More-Tab genannt) abgebildet (drei übereinander oder nebeneinander stehende Punkte), hinter dem sich weitere Aktionen befinden. Mehr zu der Action-Bar in Kapitel 8, »Von Controls, Views und Komponenten bis Atomic Design und Design-Systemen«.

7.2 Informationsarchitektur und Navigation

Die Informationsarchitektur, auch bekannt als IA, ist das Fundament Ihrer App. Eine gut entwickelte App mit einem ansprechenden ästhetischen Design wird ohne vernünftige Informationsarchitektur nicht erfolgreich sein.

Die IA ist dabei nicht gleichzusetzen mit der Navigation. Zwar hängen diese Konzepte eng miteinander zusammen, und es ist logisch, dass die Informationsarchitektur das Design der Navigation bestimmt. Aber die IA erstreckt sich weit über die App-Navigation hinaus.

Es ist eine Kunst, aus Informationen nutzbare, im Kontext sinnvolle Inhalte zu machen. Eine gute Informationsarchitektur leistet im Idealfall genau das. Eine Informationsarchitektur und die daraus resultierende Navigationsstruktur beschreiben nicht nur, wie welche Informationen strukturiert und angeordnet werden, sondern auch, wie der Nutzer damit interagiert. Das Ganze wird bei mobilen Geräten noch schwieriger, da es verschiedene Plattformen sowie Geräte gibt, die alle unterschiedlichen Interaktionsmodellen folgen.

Ebenso ist es hilfreich, sich frühzeitig mit den Entwicklern kurzzuschließen, wenn Sie eine Informationsarchitektur entwickeln, da diese später eine eigene Architektur der App herstellen müssen und es Schnittmengen in der Architektur geben kann. Änderungen an einer App-Architektur sind im Nachhinein mit größeren Anstrengungen verbunden. Das heißt, wenn Sie Ihre Informationsarchitektur derart ändern, erfolgen entsprechende Änderungen in der Datenbank.

Meist basieren Betriebssysteme auf einer hierarchischen Struktur, der klassischen Ordnerstruktur. In mobilen Apps gibt es zwar kein vollwertiges Ordnersystem, jedoch sind auch hier die Betriebssysteme hierarchisch aufgebaut. Apple bietet über die App-Files eine Ordnerstruktur an, für Android gibt es ebenso Apps, mit denen Sie Ihre Daten in einer Ordnerstruktur betrachten können.

Für Ihre App muss eine hierarchische Struktur jedoch nicht gelten. Gerade für komplexe Tablet-Anwendungen können andere Strukturen besser geeignet sein. Eine Universallösung gibt es allerdings nicht. Hier muss von Fall zu Fall entschieden werden. Egal, für welches Navigationskonzept Sie sich entscheiden, wichtig ist, dass die richtigen Inhalte den Anwender an genau der richtigen Stelle im richtigen Moment erreichen. Eine gute Informationsarchitektur beachtet also auch immer den Kontext, in dem sich der Nutzer befindet.

Als App-Designer sollten Sie einige der gängigsten Navigationsstrukturen, denen immer auch eine gründlich ausgearbeitete Informationsarchitektur zugrunde liegen muss, kennen und einsetzen können. Im Folgenden werde ich näher auf verschiedene Navigationskonzepte eingehen und Vor- und Nachteile vorstellen.

7.2.1 Der Klassiker: die hierarchische Navigation

Die Hierarchiestruktur, oder auch Baumstruktur genannt, ist eine der klassischsten Navigationsstrukturen. Eine hierarchische Navigation liegt dann vor, wenn der Anwender durch Anklicken von Links immer weiter in ein bestimmtes Thema eindringen kann. Sie basiert auf einer Hauptseite, der sogenannten Indexseite, und weiteren Unterseiten. Diesen Unterseiten können wiederum weitere Unterseiten folgen. Nutzer kennen diese Struktur sehr gut, da z. B. Ordnersysteme auf dem Desktop genauso funktionieren.

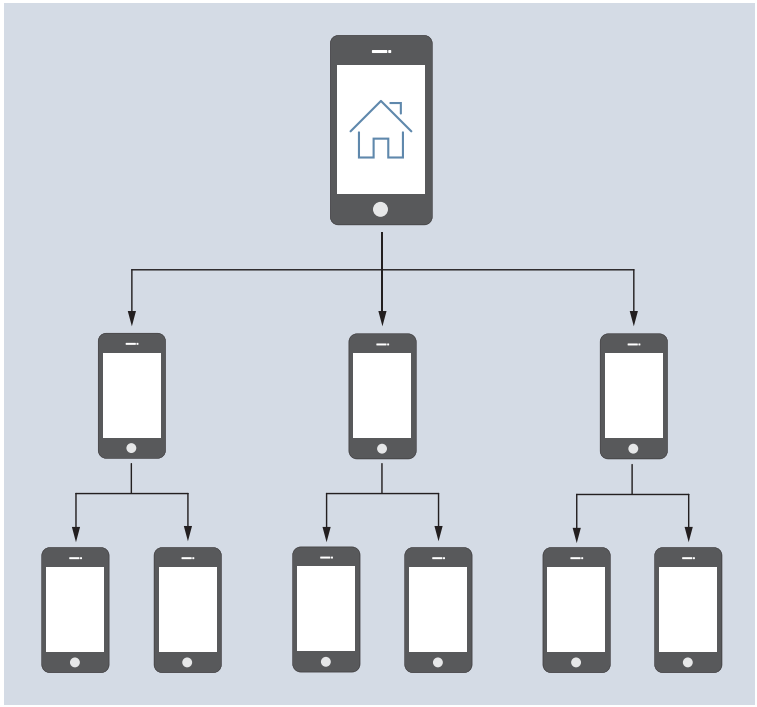

Abbildung 7.12 Schema der hierarchischen Navigation

Indexseite

Eine Indexseite ist der Startpunkt einer Applikation. Es ist der Homescreen bei einer App, entsprechend der Startseite einer Website.

Pro

- gut geeignet zur Organisation komplizierter und vor allem tiefer Strukturen
- optimal für Apps, bei denen der Nutzer in wenigen Schritten sein Ziel erreichen kann

Contra

- Problematisch bei kleinen Screens durch komplexere Inhalte oder verschachtelte Navigation. Der Nutzer kann sich recht schnell verlieren und in die falsche Richtung navigieren.

7.2.2 Jede Seite für sich: Hub & Spoke

Die Hub-&-Spoke-Navigation ist ebenfalls von einem hierarchischen Charakter geprägt. Das Pattern Hub & Spoke hat eine zentrale Indexseite, aus der der Nutzer

navigieren kann: den sogenannten *Hub*. Der Anwender sieht also nach dem Starten der App einen Hub-Screen mit Icons. Bei einer Unterseite spricht man von einem Spoke, der wiederum ein Hub sein kann. Möchte der Nutzer nun in einen anderen Spoke wechseln, muss er wieder zurück zum (letzten) Hub.

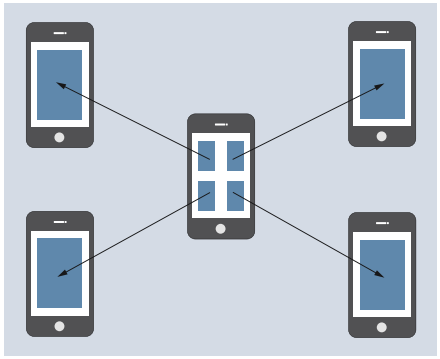

Abbildung 7.13 Schema der Hub-&-Spoke-Navigation

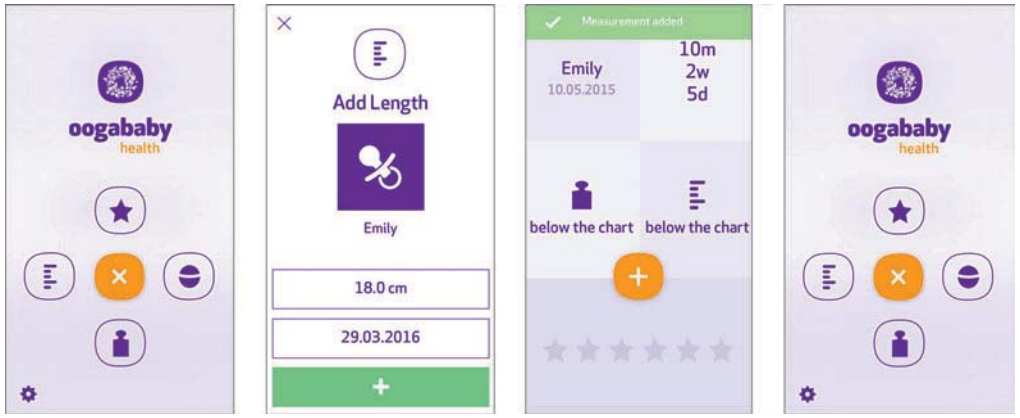

Abbildung 7.14 Hub-&-Spoke-Beispiel in der oogababy-App für iOS

Das klassische Beispiel einer derartigen Navigation sind die Homescreens aller gängigen Apps mit einer Dashboard-Funktionalität. Hier findet der Nutzer eine Übersicht all seiner installierten Apps, die er von hier aus öffnen kann. Durch die Betätigung des Home-Buttons kann der Anwender immer wieder zurück zum Homescreen gelangen.

Pro

- Multifunktionale Tools, jedes für sich mit interner Navigation und Funktion. In einem Spoke kann ein Workflow abgebildet sein, den der Nutzer über mehrere Seiten durchführen muss. Der Anwender kann immer wieder einfach zum Hub zurückkehren.

Contra

- weniger geeignet für Nutzer, die multitasken und zwischen den Spokes hin und her wechseln wollen

7.2.3 Ineinander gestapelt: Nested Doll

Die verschachtelte Matroschka-Puppe gibt dem klassischen Navigationskonzept *Nested Doll* seinen Namen. Die Nested-Doll-Struktur führt den Nutzer in einer linearen Weise zu detaillierterem Content. Der Nutzer startet in einer Listenansicht von einer Startseite, die die Hauptbereiche der App voneinander trennt. Tippt er einen Listeneintrag an, gelangt er zur nächsten Unterseite. Befindet sich der Nutzer in einer schwierigen Situation, kann er stets schnell und einfach zurücknavigieren.

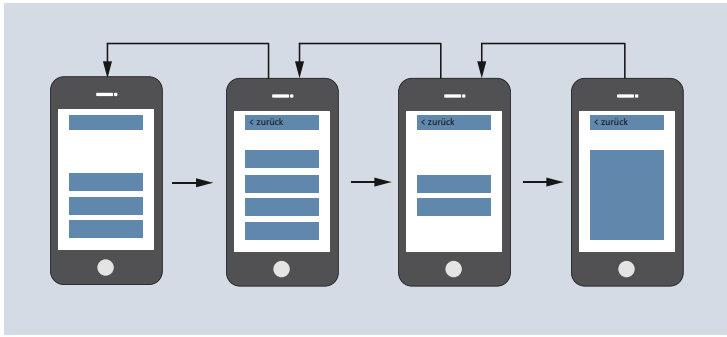

Abbildung 7.15 Schema einer Nested-Doll-Navigation

Ein gutes Beispiel sind hier Apps, die eine Timeline-Funktion haben, wie Facebook, Twitter oder Instagram. Hier gibt es aufgelistete Bereiche, durch die der Nutzer dann auf eine Unterseite gelangen kann. Dort sind weitere Details über den ausgewählten Post, Tweet oder das Bild zu finden.

Pro

- Diese Navigation eignet sich gut für Apps mit einer einfachen Aufgabe.
- Sie kann auch verwendet werden für Unterseiten in anderen Patterns.

Contra

- Der Nutzer kann nicht einfach zwischen Seiten wechseln. Je nach Tiefe eines Nested Dolls muss er entsprechend wieder alle Stufen zurückgehen, um zu einer Übersicht zu gelangen.

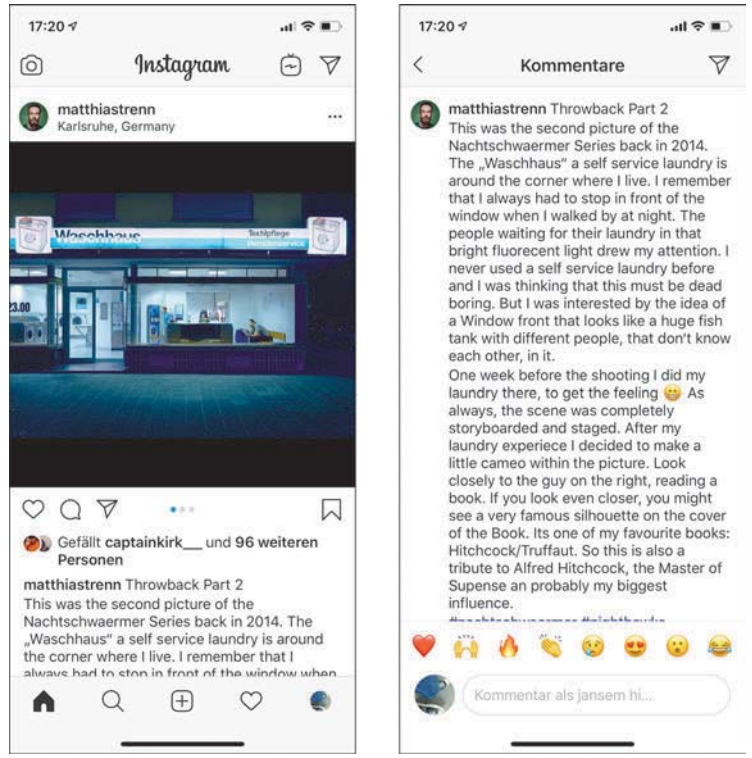

Abbildung 7.16 Nested Dolls auf Instagram

7.2.4 Für Multitasker: Tabbed View

Die Tabbed View ist wohl die bekannteste Struktur unter allen Plattformen. Sie ist die klassische Ansicht mit Tabs am unteren (iOS) oder oberen (Android) Rand des Screens.

Der Nutzer kann schnell zwischen den einzelnen Sektionen hin und her springen. In manchen Apps ist der mittlere Tab der Hauptfunktion vorbehalten, wie z. B. der Auslöser-Button in Foto-Applikationen.

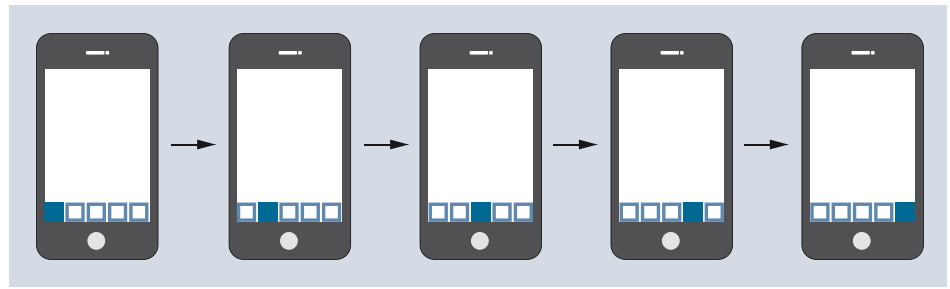

Abbildung 7.17 Schema einer Tabbed-View-Navigation

Abbildung 7.18 Onefootball-App mit einer Tabbed View

Ein gutes Beispiel für die Tabbed View ist die Onefootball-App. Im unteren Bereich des Bildschirms kann der Nutzer zwischen den verschiedenen Bereichen der App hin und her wechseln, z. B. News, Spiele, Browse, Friends, Profil.

Pro

- Ermöglicht eine schnelle Navigation und erlaubt Multitasking.
- Zudem kennt jeder Smartphone- oder Tablet-Nutzer diese Struktur und weiß auf Anhieb, wie diese zu nutzen ist. Die Struktur funktioniert also auf allen Plattformen.

Contra

- Bei zu komplexen Apps lässt sich diese Struktur nur schlecht realisieren. Die Navigation funktioniert am besten bei einfachen Informationsarchitekturen. Unter iOS sollte man nicht mehr als fünf Tabs verwenden. Sollten Sie mehrere Bereiche haben, nutzen Sie ein More-Tab und verstecken das weitere Menü unter einer neuen Listenansicht.

7.2.5 Mehr Übersicht: Bentobox/Dashboard

Die Bentobox, oder auch Dashboard genannt, bringt kleine, detaillierte Informationen auf der Indexseite unter. Im Grunde ist sie der Hub-&-Spoke-Variante sehr ähnlich. Jedoch werden hier in den Kacheln selber schon detaillierte Informationen angezeigt. Diese Variante eignet sich aufgrund der Screengröße weniger für Smartphones, sondern eher für Tablets. Sie stellt Informationen von Funktionen oder Inhalten auf einer Übersichtsseite dar. Ein Dashboard wird vorwiegend bei Tablets verwendet, auf Smartphones hingegen findet man es seltener.

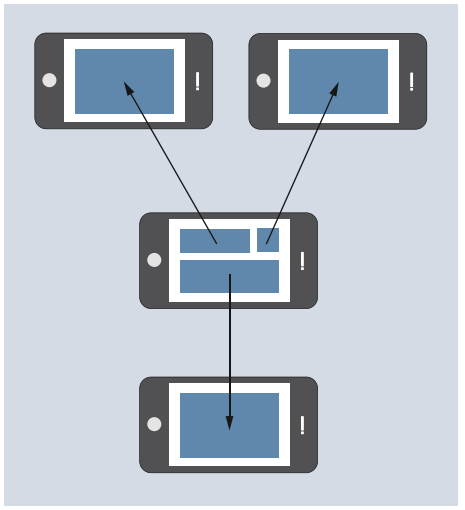

Abbildung 7.19 Schema einer Bentobox- bzw. Dashboard-Navigation

Pro

- Ein Dashboard kann sehr kraftvoll sein, denn es ermöglicht dem Nutzer, auf einen kurzen Blick alle relevanten Informationen zu sehen und primäre Aktionen zu starten.

Contra

- Die Schwierigkeit des Bentobox-Modells liegt in der Darstellung der Informationen. Es muss auf ein gut durchdachtes grafisches Interface-Design geachtet werden.

7.2.6 Der Contentking: Navigation durch Filter

Sollte der Nutzer mit einem großen Umfang von Inhalten arbeiten, empfiehlt sich die sogenannte Filtered View. Zwar ist die Filtered View kein reines Navigationskonzept, sondern besser als Inhaltselement zu verstehen, doch kann sie Nutzer bei der Navigation unterstützen.

Die Filtered View hilft den Nutzern dabei, mittels Aktivierung von Filtern den Inhalt zu konkretisieren. Sie hilft den Nutzern dabei, gezielt nach Inhalten zu suchen bzw. sie zu finden. Hierbei wählt der Nutzer aus einer Reihe von Filtern den gewünschten aus und bekommt sofort geordnete Ergebnisse angezeigt.

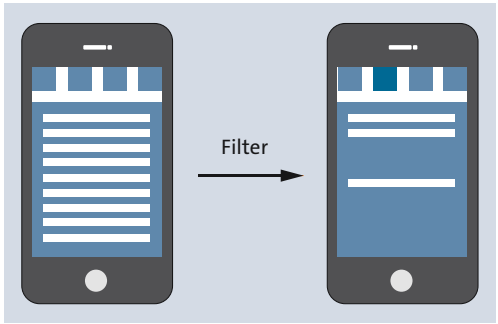

Abbildung 7.20 Schema einer Filtered View

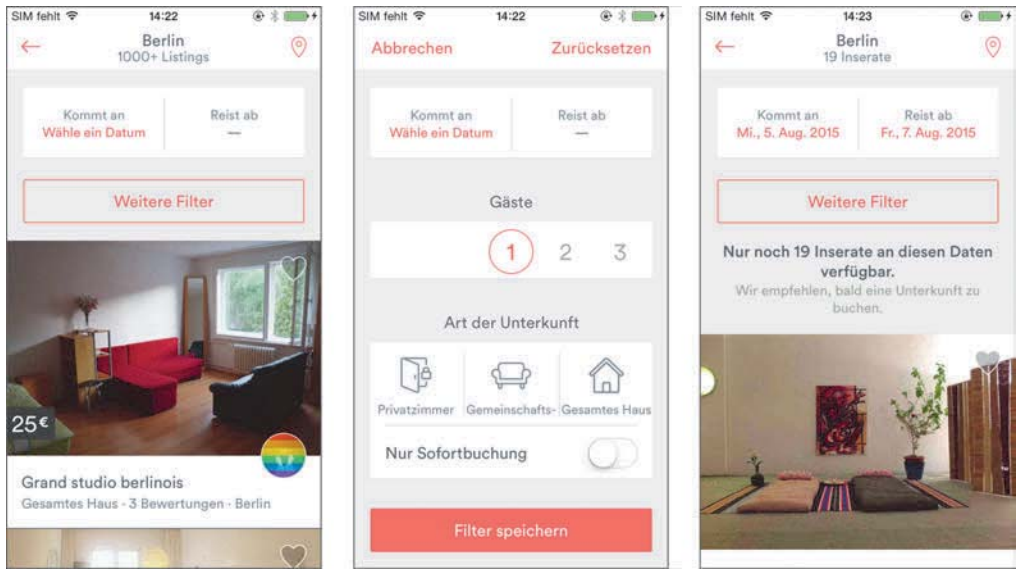

Abbildung 7.21 Filtered View in der Airbnb-App, in der Filtereinstellungen möglich sind. Interaktiv werden Ergebnisse ausgeblendet und nach Setzen des Filters angezeigt.

Pro

- geeignet für Apps mit umfangreichen Inhalten wie Artikel, Bilder oder Videos
- gut geeignet für Shopsysteme auf Tablets
- Funktioniert sehr gut als Unterseite einer der anderen Navigationsstrukturen.

Contra

- Es kann sehr schwierig sein, die Filter auf Smartphones oder Wearables ordentlich darzustellen. Meist sind diese selbst eine Liste, und die Bildschirmgröße reicht meist nicht aus.

7.2.7 Exkurs: Navigation auf Wearables

Über Apple Watch OS Glances hat die App einen kleinen Bereich, in dem Informationen der App angezeigt werden können. Der Nutzer gelangt über diese Glances in die installierte Apple-Watch-App und kann weitere Aktionen ausführen. Man kann die Glances-Ansicht mit der Mitteilungszentrale für das iPhone bzw. iPad vergleichen. Als Beispiel sei die App *Lifesum* genannt (Abbildung 7.22).

Bei *Lifesum* handelt es sich um eine Kalorienzähler-App. Die App zeigt dem Anwender seinen aktuellen Kalorienverbrauch und seine Kalorienaufnahme an. Bei Tap auf die Lifesum-Glance gelangt der Nutzer in die Apple-Watch-App, wo er weitere Aktionen ausführen kann.

Abbildung 7.22 Lifesum-App auf dem Smartphone (links) und die Apple-Watch-App (rechts)

Pro

- Die Dashboard-Navigation ist auf Wearables vor allem sinnvoll, wenn umfangreiche Applikationen vorliegen, die eine Übersichtsseite benötigen. Vorwiegend wird das Dashboard bei Applikationen genutzt, die statistische Auswertungen machen sowie Diagramme oder Kennzahlen darstellen möchten.

Contra

- weniger geeignet für kleinere Applikationen, die keinen großen Funktionsumfang besitzen

7.2.8 Der Kombinator: Navigationskonzepte miteinander kombinieren

Wie Sie vielleicht schon erkannt haben, können die verschiedenen Navigationsstrukturen miteinander kombiniert werden. Versuchen Sie, maximal zwei Navigationsstrukturen zu nutzen. Kombinieren Sie zwei Navigationsstrukturen, sollten Sie mit einer Hauptnavigationsstruktur arbeiten und die andere Struktur nur untergeordnet verwenden.

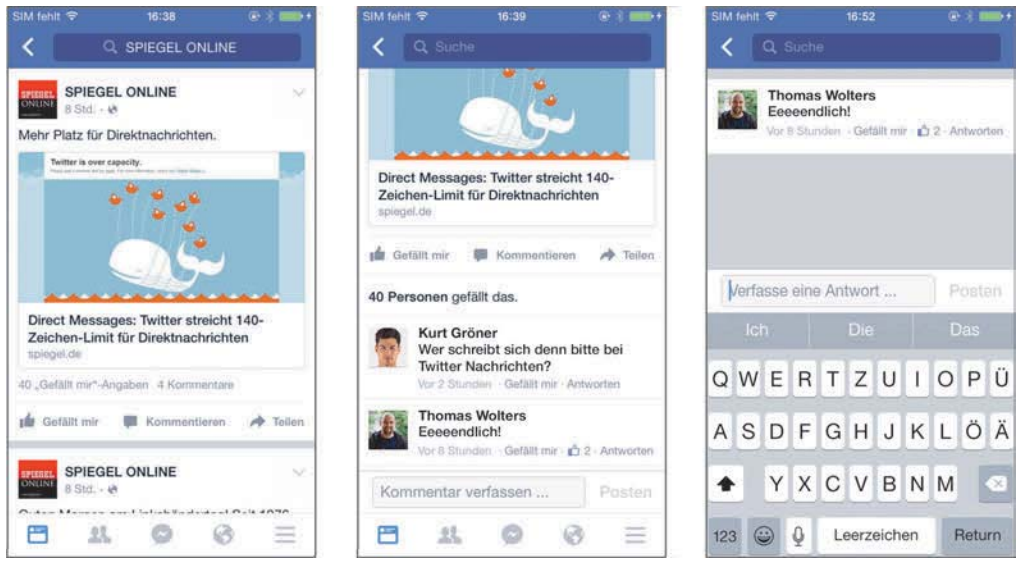

Abbildung 7.23 Eine alte Version der Facebook-App mit einer Nested-Doll-Struktur

In der Facebook-App finden Sie z. B. eine Kombination von Navigationsstrukturen. Elemente aus der Timeline verhalten sich nach Regeln der Nested-Doll-Navigation, die Hauptnavigation der App folgt allerdings der Tabbed View. In der Tabbar werden alle Hauptseiten der Applikation angezeigt, der Nutzer kann hier zwischen den einzelnen Bereichen hin und her wechseln.

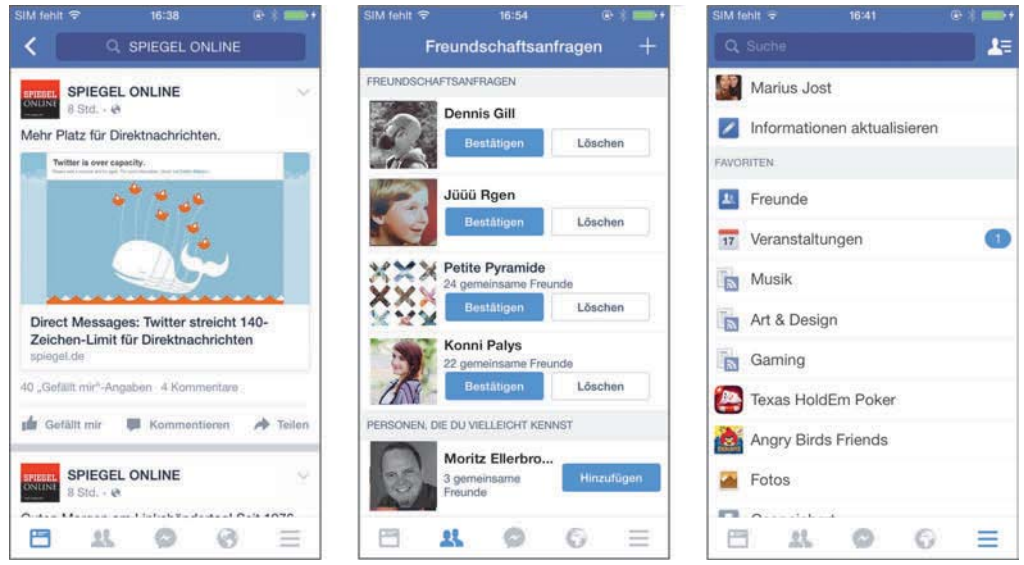

Abbildung 7.24 Eine alte Version der Facebook-App mit einer Tabbed View

Pro

- Eine Kombination verschiedener Navigationskonzepte bietet sich bei komplexen Applikationen an, wenn verschiedene Werkzeuge und Inhalte miteinander kombiniert werden müssen.

Contra

- Navigationskombinationen sollten nicht erstellt werden, um Probleme im Konzept oder in der Informationsarchitektur zu lösen.

Checkliste: Konventionen beachten

- Adaptieren Sie keine iOS-Prinzipien auf die Android-Plattform oder umgekehrt.
- Reduzieren Sie den Cognitive Load. Unterschiedliche Elemente für die gleiche Funktion erschweren die Nutzung Ihrer App. Simplifizieren Sie!
- Erarbeiten Sie sich eine Informationsarchitektur. Diese stellt die Basis für die Navigation in Ihrer App dar und hilft dem Nutzer, Bereiche zu verstehen.
- Erlernen Sie die verschiedenen Informationsarchitekturen und Navigationsarten, und wenden Sie diese an.

In diesem Kapitel haben Sie einiges zur Informationsarchitektur von Apps gelesen. Im folgenden Kapitel werde ich Ihnen die verfügbaren Komponenten auf den beiden Plattformen vorstellen und zeigen, wofür diese gedacht und auch genutzt werden sollten.

Kapitel 13

Icons, Grafiken und Bilder

Die Verwendung von Icons, Grafiken und Bildern in einer App ist neben der typografischen und farblichen Gestaltung ein weiteres wichtiges Designmerkmal. Während Icons Funktionen visuell erklären, sind Bilder oder Grafiken ein Träger von Emotionen. In diesem Kapitel gehe ich auf diese grafischen Elemente näher ein und beleuchte, wie sie in der Gestaltung richtig eingesetzt werden.

Der Einsatz von Icons, Grafiken und Bildern ist eines der wichtigsten Designmerkmale im App-Design und bestimmt stark das Branding und somit die Identität einer App. Es spiegelt die Marke wider und verkörpert die Philosophie des Unternehmens. Ebenso sind Icons, Grafiken und Bilder wesentlicher Teil des User Interfaces einer App, da sie den größten gestalterischen Teil ausmachen.

Bilder verbinden Menschen

Seit Smartphone & Co. kann jeder immer und überall Bilder machen, die anschließend häufig mit einer größeren Öffentlichkeit geteilt werden. Es ist kein Zufall, dass sich soziale Netzwerke, die sich auf visuelle Inhalte spezialisiert haben, großer Beliebtheit erfreuen: Instagram, Pinterest, Snap oder Vine leben (gut!) von Bildern und Videos.

Bilder wirken emotional. Sie sind eines der effektivsten Mittel, Emotionalität herzustellen, und eignen sich hervorragend, um eine App anschaulicher und auch persönlicher zu gestalten.

Ohne weitere Worte und Beschreibungen schaffen es Bilder, unsere Aufmerksamkeit zu gewinnen. Sie sprechen uns emotional an – direkter und viel schneller, als es Text je könnte. Das liegt daran, dass die Aussage eines Bildes in der Regel innerhalb eines Bruchteils einer Sekunde transportiert und wahrgenommen werden kann.

Bilder können auch dazu genutzt werden, App-Nutzer in eine bestimmte Stimmung (z. B. in Kauflaune/Kontaktlaune) zu versetzen. Geschickt eingesetzte Produktfotos stimulieren unser Gehirn und erzeugen das Gefühl des »Habenwollens«. Dazu später in Abschnitt 13.3.3, »Produktfotografien«, mehr.

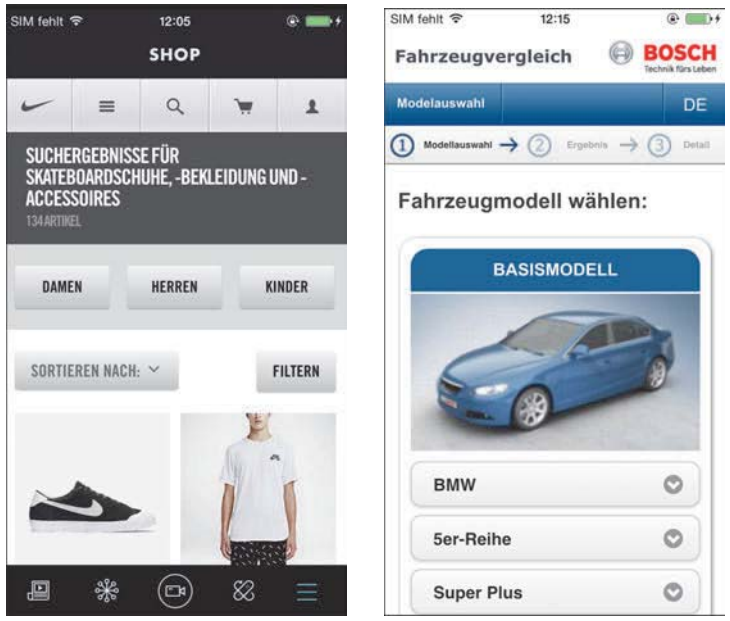

Abbildung 13.1 Eine Nike-App tritt mit einer klaren visuellen Markensprache auf. Die Bosch-App hingegen spiegelt nicht die visuelle Markensprache des Unternehmens wider, sondern bietet ein beliebiges, standardisiertes Design.

Bei der Gestaltung mit Bildern sollte aber ebenso wie bei der Auswahl der Farbwelt oder beim Einsatz der Typografie immer die Funktion der App im Vordergrund stehen.

Bilder bzw. Grafiken können in drei Bereiche unterteilt werden:

- Icons: dienen vor allem dem Interface
- Grafiken/Visuals: unterstützen die User Experience
- Fotografien: dienen als Träger von Emotionen

13.1 Icons

Icons sind kleine Symbole, die eine Aussage auf einen Blick vermitteln können und die unabhängig von Sprache und Kultur verständlich sind. Daher eignen sie sich hervorragend für das App-Design. Im App-Design sind Icons nicht mehr wegzudenken und gerade bei Navigationskonzepten ein wichtiges Hilfsmittel bei der Gestaltung.

Abbildung 13.2 Bahnübergangssymbol im deutschen Straßenverkehr

Man unterscheidet unter anderem zwischen symbolischen und ikonischen Icons. Grundsätzlich sollten Icons ohne zusätzlichen Text verstanden werden können. Typischerweise sind User-Interface-Icons symbolisch oder ikonisch, aber viele liegen irgendwo dazwischen.

Symbolische Zeichen stellen ein Konzept in etwas abstrakter Weise dar. Ein Beispiel wäre das Refresh-Icon, das eigentlich in jedem Browser zu finden ist. Symbolische Zeichen können ohne Vorwissen und Kontext schwer in der Bedeutung erkannt werden. Der Nutzer muss die Bedeutung dieser Zeichen erst erlernen.

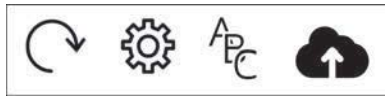

Abbildung 13.3 Beispiel für symbolische Icons

Ikonische Zeichen können Konzepte in einer fast wörtlichen Weise darstellen. Die Bedeutung kann in der Regel durch das Allgemeinwissen erschlossen werden. Ein Beispiel hierfür wäre ein Mülleimer. Der Nutzer kann erschließen, dass er über dieses Icon ein Element/einen Eintrag löschen kann.

Abbildung 13.4 Beispiele für ikonische Icons

Nach Möglichkeit sind ikonische Zeichen aufgrund ihrer eindeutigen Bedeutung einem symbolischen vorzuziehen. Versuchen Sie immer, eine visuelle Repräsentation der darzustellenden Funktion zu finden. Nutzen Sie symbolische Zeichen nur dann, wenn es keine physische Repräsentation gibt.

13.1.1 Von guten und schlechten Icons

Icons sind immer Teil des User Interfaces. Icons lassen Ihre App freundlich, einladend und professionell wirken. Aber positionieren Sie nicht einfach Icons, um eine Fläche zu füllen. Icons dienen einem bestimmten Zweck: Sie unterstützen das User Interface, sie sind aber nicht Hauptbestandteil.

Warum sollten Sie Icons nutzen?

Im App-Design geht es immer um Kommunikation. Wenn der Nutzer auf einem kleinen Bildschirm die Inhalte betrachtet, erkennt er Icons schneller als den umgebenden Text. Icons müssen dabei nicht den Text ersetzen, sondern unterstützen. Icons nutzen denselben psychologischen Zweck wie ein Zeilenumbruch in einem Text – sie brechen den Inhalt auf. Das Auge wandert automatisch dorthin.

Abbildung 13.5 Airbnb-App in einer älteren Version mit schön aufgeräumtem UI, und die Icons unterstützen das schnelle Erfassen der Inhalte durch den Nutzer.

Weißraum

Bei Weißraum (auch White-Space genannt) handelt es sich – ganz einfach definiert – um leere Fläche. Weißraum muss nicht zwangsläufig weiß sein. Daher wird er auch als »negativer Raum« oder »Leerraum« bezeichnet und ist der Freiraum zwischen den Elementen, ganz unabhängig von der farblichen Gestaltung. Ausreichend freie Fläche bzw. Weißraum ist ein wichtiger Bestandteil jedes guten Layouts. Als Gestalter arbeiten sie mit dem Weißraum genauso intensiv wie mit den Texten oder anderen Objekten Ihres Designs. Er wird benötigt, um Abstände zwischen Elementen zu schaffen, und sorgt für Übersichtlichkeit und Hierarchie.

Das primäre Ziel eines Icons sollte es sein, den Nutzer effizient Inhalte erkennen und verarbeiten zu lassen. Meist wird dies erreicht durch die gelungene Gestaltung mit Weißraum und die Verwendung von Icons, die den Inhalt nicht stören, sondern unterstützen. Die Vorteile von Icons in User Interfaces sind:

- Icons sind gute Touchziele. Sie sind groß genug, um eine Touchfläche zu sein.
- Icons haben eine Schutzzone (Weißraum) um sich, in ihrer kompakten Darstellungsweise können sie gut aneinandergereiht werden.

- Icons können schnell durch einen kurzen Blick erkannt werden, vorausgesetzt, sie sind eindeutig und gut gestaltet. Das funktioniert in der Regel mit etablierten Icons, die der Nutzer bereits kennt, wie z. B. einer Lupe.
- Icons können international und unabhängig von einer Kultur verstanden werden. So wird ein Briefumschlag international als Symbol für E-Mails verstanden.
- Icons können optisch ansprechend sein und zur Verbesserung der Ästhetik eines Designs beitragen.
- Icons können, wenn sie absolut eindeutig sind, als Ersatz für Texte dienen.
- Icons vereinfachen die Verbreitung der App in unterschiedlichen Sprachen. Die Sprachanpassungen werden minimiert und etwaige Probleme in der Gestaltung damit gelöst: Ein französisches Wort passt vielleicht nicht ins Layout, die deutsche Übersetzung aber vielleicht schon …

Icons können allerdings auch Usability-Probleme verursachen, wenn sie ohne Rücksicht verwendet werden.

Die Verwendung von Icons

Icons sollten immer einen Zweck kommunizieren und z. B. dort verwendet werden, wo Aktionen ausgeführt werden sollen. Icons sind per Definition eine visuelle Darstellung eines Objekts, einer Aktion oder einer Idee. Wenn das Objekt, die Aktion oder die Idee dem Nutzer nicht klar vermittelt wird, sondern lediglich als ästhetisches Element dient, erzeugt dies »visuelles Rauschen« und damit Frustration.

Visuelles Rauschen

Ein visuelles Rauschen, das z. B. durch zu viele Aufmerksamkeitseffekte und einen Überschuss an visuell Verschiedenem ausgelöst werden kann, ist grundsätzlich zu vermeiden.

Es gibt Icons, die universell sind. Das heißt, sie werden von den meisten Nutzern direkt erkannt. Eine Lupe z. B. wird von Nutzern mit einer Suchfunktion in Verbindung gebracht, während einem Drucker-Icon eine Druckfunktion zugeordnet wird.

Abgesehen von diesen Beispielen gibt es Icons, die für Funktionen stehen, die nicht eindeutig vom Nutzer verstanden werden. Es gibt keine festen Regeln für Icons. Der Nutzer weiß nicht zwangsläufig, welche Funktion hinter einem Icon steckt. Erst durch Nutzung des Interfaces werden Icons nach und nach verstanden. Ein gutes Beispiel hierfür ist das sogenannte Burger-Icon. Dieses Icon wird mittlerweile von vielen in seiner Funktion erkannt. Ein ähnliches Icon sind die sogenannten Elipses oder auch More-Dots genannt. Dies finden Sie vorwiegend in Android-Apps. Es steht in der Regel für ein Hauptmenü einer App oder sammelt weitere Bereiche oder Funktionen in sich. Jedoch gibt es auch Apps, die diese Funktion mit einem Liste-Icon symbolisieren.

Abbildung 13.6 In der TVPro-App (links) werden sehr missverständliche Icons eingesetzt, ein Uhr-Icon für »Liste«. Die Deezer-App (rechts) macht das schon besser.

Ein weiteres Beispiel ist die Verwendung von Herz- und Stern-Icons. Diese Icons werden verwendet, um Inhalte zu favorisieren, zu bookmarken oder zu bewerten. Der Einsatz variiert von App zu App. Bestes Beispiel: Twitter. Hier wurde »Favorisieren« 2015 von einem Stern in ein Herz verwandelt (und damit auch die Funktionsweise angepasst). Mehr dazu unter *www.theverge.com/2015/11/3/9661180/twitter-vine-favorite-fav-likes-hearts*. Nun repräsentiert es ein Like statt einer Favorisierung. Zwei komplett unterschiedliche Funktionen.

Falsch eingesetzte Icons können den Nutzer in die Irre führen und damit verwirren und frustrieren. Verwenden Sie Icons, die klar und eindeutig die Funktion, die dahintersteckt, symbolisieren. Bei symbolischen Icons, die einen Refresh-Button oder eine Cloud-Funktion darstellen sollen, orientieren Sie sich an bereits etablierten Symbolen.

Icons funktionieren gut in Navigatoren und helfen dabei, Bereiche zu definieren und ein Symbol für ebendiese festzulegen. Ein kleiner Tipp von mir, sollten Sie keinen guten Einfall für ein Icon haben: Ersinnen Sie ein Verb oder einen Begriff des zu suchenden Icons, und nutzen Sie die Google-Bildersuche. Die Suchergebnisse geben Ihnen eine Hilfestellung bei der Wahl des richtigen Icons.

13.1.2 Arten von Icons

Es gibt verschiedene Arten von Icons. Während Icons vor einigen Jahren noch sehr detailliert und realistisch (Stichwort Skeumorphismus) gestaltet und bunt eingefärbt

wurden, sind sie derzeit eher flach und reduzierter und insgesamt funktioneller gehalten. Es haben sich zwei Icon-Arten im App-Design etabliert: Stroked und Filled Icons, auch Glyphen genannt.

Stroked Icons

Stroked Icons sind Icons, die aus Konturen bestehen. Sie zeichnen sich durch einen höheren Detailgrad aus als Filled Icons (siehe den nächsten Abschnitt). Sie wirken filigraner und nehmen weniger Raum ein. Sie eignen sich hervorragend für Apps mit detailliertem Design.

Abbildung 13.7 Beispiele für Stroked Icons

Bei Stroked Icons gibt es allerdings Skalierungsprobleme. So weisen skalierte Icons eine größere Strichstärke auf, was zu einem ästhetischen Bruch führt.

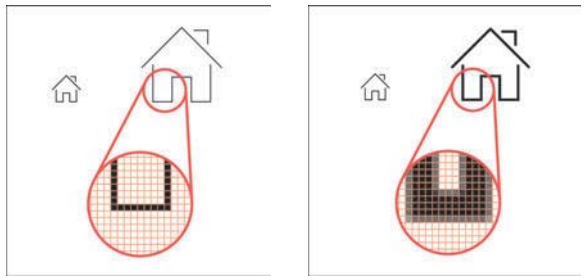

Abbildung 13.8 Links: gleichbleibende Konturstärke bei unterschiedlichen Größen des Icons, rechts: unterschiedliche Konturstärke bei unterschiedlichen Größen des Icons

Bei der Verwendung dieser Icons sollten Sie darauf achten, dass jedes Icon beim Skalieren ggf. die Strichstärke mitskaliert. Für größere Objekte sollten Sie die Strichstärke anpassen, um ein ästhetisch harmonisches Gesamtbild zu erhalten.

Filled bzw. Glyph Icons

Filled Icons oder auch Glyphen genannt sind einfarbige Symbole, die durch eine Fläche entstehen. Hierbei nimmt das Symbol mehr Raum ein als ein Stroked Icon. Filled Icons haben meist weniger Details, sind aber, wenn sie gut gestaltet sind, sehr einfach zu erkennen.

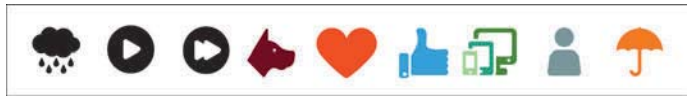

Abbildung 13.9 Beispiele für Filled Icons

Sie eignen sich jedoch hervorragend für einfache Apps, die wenige, aber recht plakative Icons benötigen. Der Vorteil von Filled Icons ist, dass sie keinem Verhältnisfaktor unterliegen, d. h., sie können skaliert werden, ohne dass es einen ästhetischen Bruch gibt. So reicht eine definierte Größe über alle Icons hinweg vollkommen aus.

Filled oder Stroked?

Es gab Untersuchungen (von Curt Arledge von Viget, *https://viget.com/inspire/are-hollow-icons-really-harder-to-recognize-a-research-study*), welcher Stil von Icons schneller vom Nutzer erfasst wird – einen klaren Gewinner gab es hierbei aber nicht.

Grundsätzlich wurde nur bestätigt, dass ein gut gestaltetes Icon, das mit seiner Form die dahinterliegende Funktion deutlich repräsentiert, schneller erkannt wird als Icons, bei denen das nicht der Fall ist. Dabei ist es jedoch unerheblich, welchen Stil das Icon hat. Seien Sie in Ihrer Gestaltung aber konsistent, und wechseln Sie nicht zwischen Filled und Stroked Icons hin und her. Icon-Stile sollten nicht miteinander vermischt werden, denn es erschwert dem Nutzer das Erfassen, da er bei einen Symbol nach bestimmten Charakteristiken sucht. Eine Ausnahme bietet folgendes Szenario: Die App verfügt über eine Favoriten-Funktion. Im Header ist ein Stroke-Stern, der mir symbolisiert, dass ich mit einem Tap die Seite/ein Produkt oder Ähnliches als Favorit speichern kann. Ist die Seite/das Produkt als Favorit gespeichert, so erscheint dieser Stern gefüllt (filled).

Für beide Icon-Arten gilt: Wählen Sie nach Möglichkeit ein ikonisches Icon, das die tatsächliche Funktion darstellt und sich nicht nur symbolisch darauf bezieht. So wird in Abbildung 13.10 das Taschenrechner-Icon links schneller erkannt als das rechte.

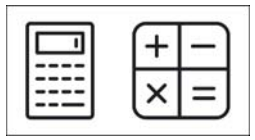

Abbildung 13.10 Taschenrechner gut (links), Taschenrechner schlecht (rechts)

13.1.3 Tipps für gelungenes Icon-Design in Apps

Wenn es an die Icon-Gestaltung Ihrer App geht, ist es sinnvoll, zuvor auch einmal die Icons von Konkurrenz-Apps oder die Ihrer Lieblings-Apps anzuschauen, um sich inspirieren zu lassen. Auch die einschlägigen Portale wie Dribbble, Behance & Co., die ich Ihnen bereits in Kapitel 10, »Inspiration und Trends«, vorgestellt habe, animieren die Vorstellungskraft.

Das App-Icon

Falls Sie sich wundern, warum in diesem Abschnitt das App-Icon nicht erwähnt wird: Da dieses Icon eine Sonderrolle in Bezug auf das Marketing und die Produktkommunikation Ihrer App einnimmt, gehe ich in Abschnitt 14.3, »Warum das App-Icon so wichtig ist«, ganz ausführlich auf die Gestaltung und Platzierung des App-Icons ein. Das App-Icon tritt in direkte Konkurrenz zu anderen Apps und sollte das Markenzeichen Ihrer App sein. Das bedeutet: Es ist ungemein wichtig, dass es sich visuell von der Konkurrenz abhebt und einen hohen Wiedererkennungswert hat.

Wenn Sie selber Icons kreieren, achten Sie darauf, sie leicht erkenn- und merkbar zu gestalten, und folgen Sie dabei am besten folgenden Regeln:

Verzichten Sie auf einen zu hohen Detailgrad

Halten Sie die Gestaltung der Icons einfach und schematisch. Reduzieren Sie die Menge an grafischen Details, und konzentrieren Sie sich auf die grundlegende Charakteristik des Symbols. Oder um hier den berühmten Grafikdesigner Kurt Weidemann zu zitieren: »Ein Zeichen ist gut, wenn man es mit dem großen Zeh in den Sand kratzen kann!«

Seien Sie bei der Gestaltung der Icons also bewusst reduziert, vermeiden Sie ebenso 3-D-Effekte, 3-D-Perspektiven und Schlagschatten an Ihren Icons. Je mehr Effekte ein Icon enthält, desto mehr Zeit benötigt der Nutzer, um das Icon zu interpretieren.

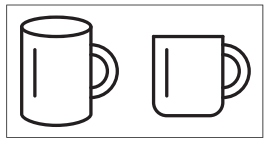

Abbildung 13.11 Links Icon mit 3-D-Perspektive und rechts ohne

Setzen Sie nur eine limitierte Farbpalette ein

Setzen Sie die Farbe für Ihre Icons bewusst und dosiert ein. Eines der Entscheidungsmerkmale bei der Farbwahl Ihres Icons ist die Farbe des Hintergrunds, auf dem das Icon liegt. Versuchen Sie, einen hohen Kontrast herzustellen, am besten einen Hell-Dunkel-Kontrast, wie ich ihn im vorherigen Kapitel angesprochen habe. Je näher die Farbe des Hintergrunds an der Farbe des Icons liegt, desto schlechter wird das Icon beim Nutzer wahrgenommen und erkannt. Beachten Sie hierbei auch Sehschwächen von älteren Nutzern oder Farbenblindheit. Nur ein hoher Kontrast zwischen Hintergrund und Icon garantiert, dass das Icon wahrgenommen wird.

Testen mit Schwarzweiß

Um zu testen, ob ein Icon funktioniert, sollten Sie Ihr Icon am besten zunächst in Schwarzweiß gestalten. Überzeugt ein Icon bereits hier, dann wird es auch in Farbe funktionieren. Ein Icon muss in der simpelsten Darstellungsform und auch immer einfarbig funktionieren.

Für Sketch, aber auch Adobe XD gibt es Plugins, bei denen Sie den Kontrast messen können. Ebenso ist es möglich, Sehbehinderungen zu simulieren und beurteilen zu können. Hier finden Sie zwei Plugins dazu:

- Contrast Analyser (Sketch): *https://github.com/getflourish/Sketch-Color-Contrast-Analyser*
- Stark (Sketch & Adobe XD): *https://getstark.co*

Legen Sie eine einheitliche Größe für Icons fest

Für den Entwickler ist es wesentlich einfacher, Ihre Designs umzusetzen, wenn Ihre Icons immer dieselbe Grundgröße haben. Eine Grundgröße wäre z. B. 16 × 16 px oder 24 × 24 px. Als einheitliche Form sollte immer ein Quadrat genutzt werden, in dem das Icon zentriert platziert wird.

Haben Icons unterschiedliche Größen, müsste der Entwickler für jedes Icon die Größe und Position definieren. Was für Icons gilt, gilt übrigens auch für Grafiken und Fotografien – auch hier sollten Sie nach Möglichkeit Einheitsgrößen festlegen. Bei Grafiken und Fotografien kann es jedoch eine komplett andere Form sein, egal ob Rechteck oder Quadrat. Detaillierter gehe ich in Kapitel 8 bzgl. des Design-Systems darauf ein.

Seien Sie in Ihrer Gestaltung konsistent

Wie bereits in Abschnitt 13.1.2, »Arten von Icons«, erwähnt, sollten Icon-Stile nicht miteinander vermischt werden. Wechseln Sie also nicht wild zwischen Stroked und Filled Icons hin und her, denn dies erschwert dem Nutzer das Erfassen der Icons und lässt die Gesamterscheinung der App nicht konsequent wirken.

Abbildung 13.12 Icons ohne konsistente Gestaltung

Abbildung 13.13 Icons mit konsistenter Gestaltung

Symbolisieren Sie die Aktion durch ein Objekt

Um ein Icon intuitiv zu gestalten, versuchen Sie, ein reales Objekt zu finden, das die Aktion am besten symbolisiert. So kann das Versenden einer Nachricht durch einen Papierflieger symbolisiert werden. Das Blatt Papier steht für die geschriebene Nachricht. Achten Sie darauf, dass Sie ein Objekt verwenden, das der Nutzer erkennt und mit einer entsprechenden Aktion verbinden kann.

Komplexe Icons benötigen ein Text-Label

Ein Text-Label an einem Icon erklärt die Funktion eines Icons klarer. Es hilft vor allem bei komplexen Funktionen, die nicht einfach ikonisch dargestellt werden können. Text-Labels sollten immer sicht- und lesbar sein.

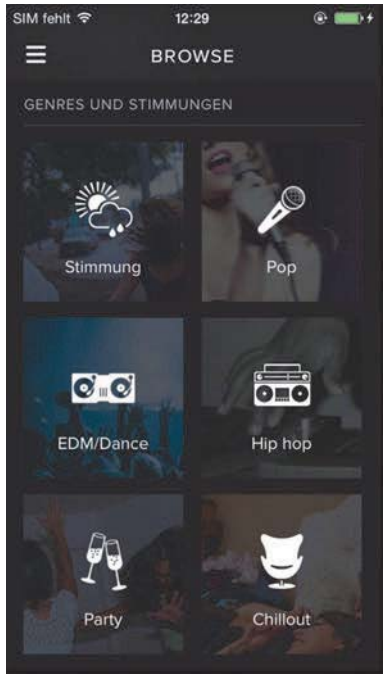

Abbildung 13.14 App mit Labels, die komplexere Icons unterstützen (hier: »Spotify«)

Vermeiden Sie die Nutzung von Icons bei abstrakten Funktionen

Das Erstellen einer E-Mail kann durch einen Briefumschlag oder einen Stift repräsentiert werden. Wie visualisiere ich aber eine abstrakte oder komplexe Funktionalität, z. B. die Prüfung eines Rauchwarnmelders über Bluetooth?

Es ist besser, Symbole für konkrete und klar verständliche Funktionen zu nutzen als für abstrakte. Bei abstrakten Funktionen sollten Sie versuchen, ein Objekt zu finden, das ansatzweise das repräsentiert, was die Funktion auch abbildet. Das Finden eines solchen Objektes ist oft nicht einfach. Erinnern Sie sich an meinen Google-Bildersuche Tipp? Nein? Dann schauen Sie doch kurz vorbei in Abschnitt 13.1.1, Seite 594.

Sehr hilfreich ist auch die Fünf-Sekunden-Regel. Wenn Sie länger als fünf Sekunden brauchen, um das Icon der gewünschten Funktion zuzuordnen, wird das Icon, das Sie gerade betrachten, nicht die Bedeutung kommunizieren.

Abbildung 13.15 Stoppzeichen im Straßenverkehr als klares Zeichen mit der Aufforderung zu stoppen

Kombination von Wort, Form und Farbe bei komplexen Icons

Für Funktionen, die nur abstrakt repräsentiert werden können, ist eine Kombination von Wort, Form und Farbe wirksamer als ein Symbol. Ein gutes Beispiel hierfür ist das Stoppzeichen.

Setzen Sie metaphorische Icons mit Bedacht ein

Seien Sie vorsichtig, wenn Sie metaphorische Icons einsetzen, da Sie Raum für Fehlinterpretationen lassen. Ein Beispiel für ein gutes metaphorisches Icon wäre eine Schere, gegebenenfalls mit dem zusätzlichen Label »Ausschneiden«. Die Schere wird von den meisten Anwendern direkt mit der Funktion »ausschneiden« assoziiert. Ein Kettensymbol jedoch, wie es häufig für Internetlinks steht, könnte missinterpretiert werden. Ein Kettensymbol repräsentiert zuerst einmal eine Kette, also ein verbindendes Element. In einer Liste könnte man das Kettensymbol so interpretieren, dass es z. B. Tabellenzeilen miteinander verbindet.

Halten Sie die Platzierung und die Reihenfolge der Icons konsistent

Nutzer ziehen nicht nur durch die Gestaltung eines Icons Rückschlüsse auf die dahinterliegende Funktion, sondern orientieren sich auch an der Position und Reihenfolge der Icons.

Wenn eines der Icons in einer späteren Version Ihrer App eine andere visuelle Erscheinung hat, wird der Nutzer aller Wahrscheinlichkeit nach dem Icon dennoch die bekannte Funktion wieder zuordnen. Wird das Icon jedoch woanders positioniert, wird es schwieriger; der Nutzer kann sich nicht mehr richtig orientieren. Daraus kann man schließen, dass die Positionierung und die Reihenfolge wichtiger als die visuelle Erscheinung sind. Verändern Sie also nicht einfach die Position und die Reihenfolge der Icons, wenn Sie das Interface neu gestalten oder neue Aktionen hinzufügen müssen.

Icons gruppieren

Icons bzw. Aktionen, die in ähnlicher Weise funktionieren, sollten gruppiert werden. Auf diese Weise hat der Nutzer die Möglichkeit, alle Optionen für seine Aufgabe gesammelt zu finden und zu nutzen. Beispielsweise könnten Sie Editierfunktionen für Text wie Bold, Kursiv, Schriftgröße und Schriftart in einer Gruppe sammeln. So findet der Nutzer alle relevanten Funktionen beisammen.

Abbildung 13.16 Sobald Icons bzw. Funktionen nach Ähnlichkeit oder Funktionalität gruppiert werden, lassen sie sich einfacher identifizieren.

Icons mit ähnlicher Funktion nicht nebeneinander platzieren

Platzieren Sie Icons mit ähnlichen Konzepten wie z. B. Aufgaben, Postfach, Benachrichtigungen nicht nebeneinander – dadurch wird die Unterscheidbarkeit der Funktionen und das Erfassen durch den Nutzer verringert.

Abbildung 13.17 Links ein gutes Beispiel für eine Side-Navigation, bei der die Icons untereinander angezeigt werden. Rechts ein negatives Beispiel – hier werden die Icons ohne klares System nebeneinander angeordnet.

Icons gleicher Art sollten ein gemeinsames visuelles Motiv haben

Icons derselben Art sollten auch ein gemeinsames visuelles Motiv haben. Ein gutes Beispiel sind die Icons bei Text-Editoren. Hier steht das **B** für einen fett geschriebenen Text, das *K* für einen kursiven Text, das U für einen unterstrichenen Text. Alle drei Icons repräsentieren unterschiedliche Funktionen, jedoch bestehen alle Icons aus einem Buchstaben, dieser ist das visuelle Motiv. Dies hilft dem Nutzer herauszufinden, welche Symbole sich ähnlich verhalten.

Testen Sie Ihre Icons

Testen Sie das Icon. Fragen Sie Kollegen oder Freunde, was sie hinter dem Icon vermuten. Testen Sie Ihre Icons auf Ihre Einprägsamkeit hin. Fragen Sie Nutzer, die Ihre App im Teststadium genutzt haben, welches Icon welche Funktion repräsentiert.

Es gibt viele Vorteile, wenn Sie Icons in Ihrem User Interface nutzen. Ein Vorteil ist, dass sie Platz sparen. Der größte Vorteil ist jedoch, dass, wenn man Icons richtig ein-

setzt, der Nutzer schneller und leichter das Interface erlernt und die App intuitiver nutzen kann. Nutzer werden nicht von Icons profitieren, wenn diese nicht klar und Intuitiv sind; das erreichen Sie nur, wenn Sie die oben genannten Regeln beherzigen.

13.1.4 Vorgefertigte Icon-Sets

Auf den verschiedenen Portalen wie Dribbble oder Behance finden Sie viele kostenfreie kleinere wenig umfangreiche Icon-Sets. Alternativ können Sie meist umfangreichere Icon-Sets käuflich erwerben, die oft bereits verschiedene Einheitsgrößen enthalten.

Gute Anlaufstellen sind auch verschiedene Portale, bei denen Sie größere Iconsets kaufen oder kostenfrei herunterladen können:

- Auf *Nucleo* können zahlreiche Icon-Sets erworben werden. Zudem können Sie Icons in verschiedenen Ausführungen (Stroked, Filled) in Projekten archivieren. Diese Icons können dann für das PNG-, JPG- und SVG-Format in verschiedenen Größen und Farben exportiert werden. So können Sie eine komplette Bibliothek für ein Projekt aufbauen und behalten den Überblick über die eingesetzten Icons. Webseite: *nucleoapp.com*.

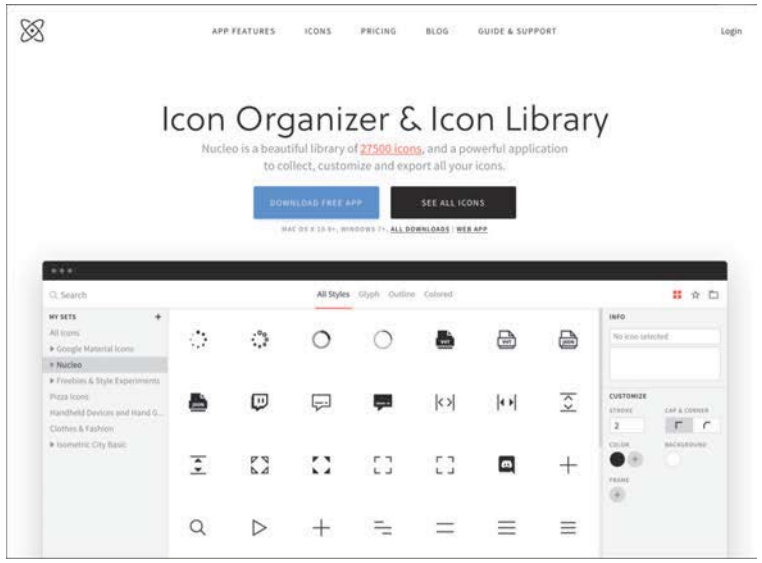

Abbildung 13.18 Nucleo

- *The Noun Project* ist eine praktische Icon-Quelle für Designer, die eine umfangreiche Sammlung kostenloser Icons und Piktogramme für nahezu jedes Themengebiet bietet. Alle auf The Noun Project angebotenen Piktogramme sind kostenlos nutzbar und stehen als Public Domain oder unter der CC-Lizenz zur Verfügung. Webseite: *thenounproject.com*.

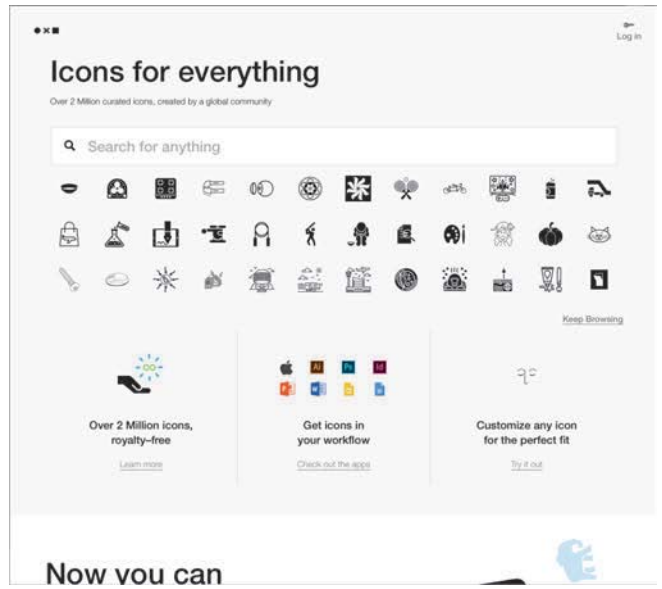

Abbildung 13.19 Webseite Noun Project

- *Streamline Icons* ist ein sehr umfangreiches Set, welches Icons in verschiedenen Ausprägungen anbietet. So gibt es Stroked Icons in verschiedenen Stufen und ebenso Filled Icons. Streamline Icons bietet zudem auch ein Online-Tool zur Verwaltung dieser Icons an, ergänzend auch ein Datei-Format für *IconJar*. Webseite: *https://streamlineicons.com*.

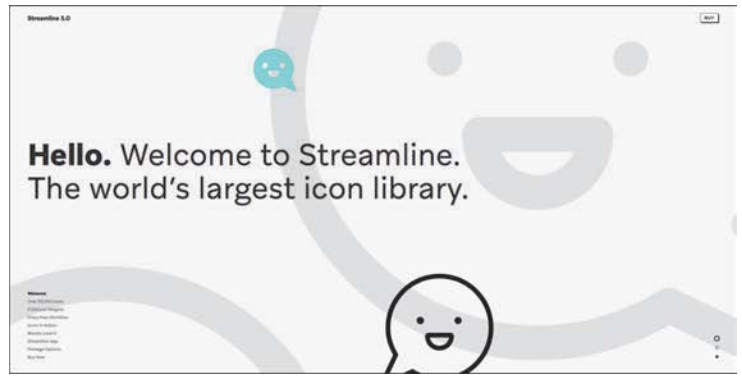

Abbildung 13.20 Streamline-Icons-Webseite

- *iconmonstr* bietet ebenfalls kostenlose Icons und Piktogramme im PNG- und SVG-Format an. Alle Icons sind einfarbig und stark stilisiert bzw. simpel gestaltet. Webseite: *iconmonstr.com*.

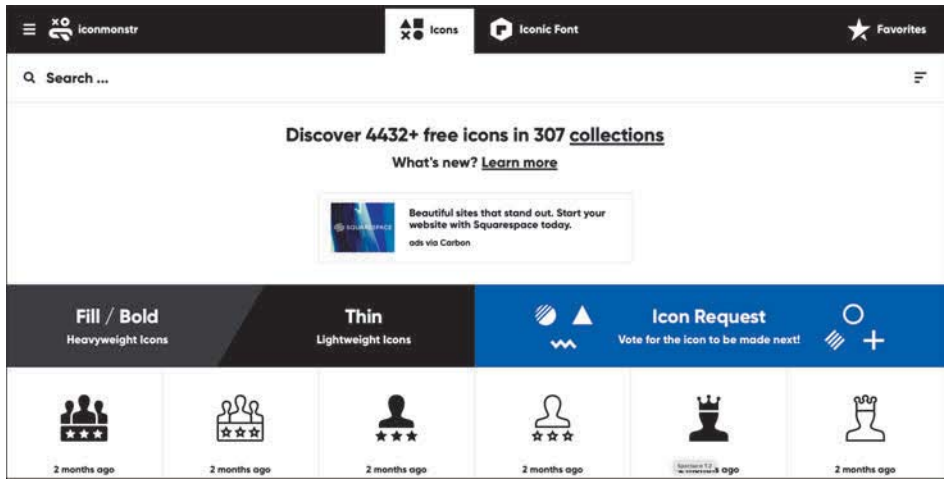

Abbildung 13.21 iconmonstr

Zuletzt möchte ich Sie noch auf ein Tool aufmerksam machen, das ich persönlich nahezu täglich nutze. Es ist das Tool *IconJar*. IconJar ist ein Icon- Verwaltungstool. In diesem können Sie Ihre Icons importieren und verwalten. Der große Vorteil ist, dass Sie per Suche Icons sehr einfach finden und per Drag & Drop in Ihren Grafikprogrammen platzieren können. Es vereinfacht die Arbeit ungemein. Webseite: *https://geticonjar.com*.

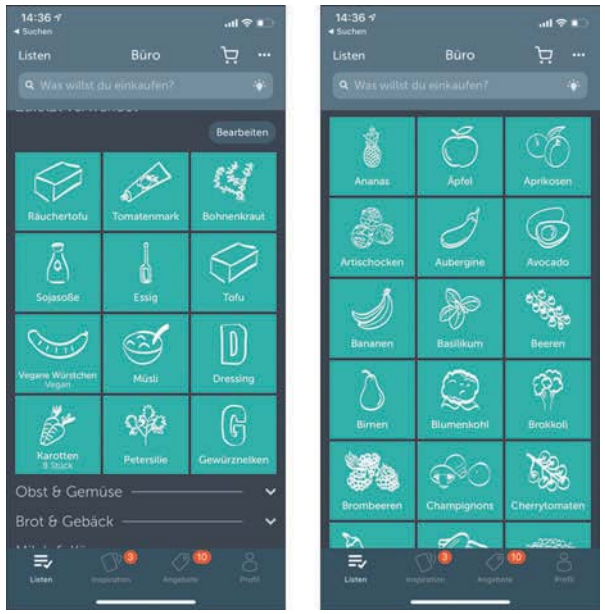

Abbildung 13.22 In der Bring!-App zeigen verständliche Icons, welche Produkte andere App-Nutzer für eine Party mitbringen sollen.

13.2 Grafiken

Grafiken oder Illustrationen sind, wenn man so will, der große Bruder der Icons. Sie sind Illustrationen, die speziell zu bestimmten Themenbereichen gestaltet werden. Während Icons einen entscheidenden Teil des App-User-Interfaces einnehmen und auch wichtig für die Bedienung und Navigation der App sind, sind Grafiken bzw. Visuals nahezu reine Gestaltungselemente und eher als Teil der User Experience zu sehen. Oft findet man solche Visuals im On-Boarding-Prozess einer App.

Grafiken gibt es in vielen Ausprägungen und Stilrichtungen. So sind einige klar als »Flat Design« zu identifizieren und andere als skeuomorphisch. Eine andere Stilrichtung ist der sogenannte Low-Poly-Stil.

Abbildung 13.23 Die Grafik einer Weltkugel in drei unterschiedlichen Stilen: skeuomorphisch bzw. realistisch, Flat Design bzw. minimalistisch und als Low-Poly-3D-Variante

Abbildung 13.24 In der Wetter-HD-Lite-App ist das Hintergrundbild Teil der Funktion: Hier werden je nach Wetter und Tageslage verschiedene Visuals (Wolke mit und ohne Sonne) gezeigt, die das Wetter veranschaulichen.

Im Idealfall ist eine Grafik in der App der große Bruder der Icons. Grafiken sind im Vergleich zu einem Icon in der Regel detaillierter und oftmals auch flächiger und meist mehrfarbig gestaltet. Natürlich sollten die verwendeten Icons und Grafiken in

einer App zueinander passen und denselben oder zumindest ähnlichen visuellen Stil verfolgen.

Wichtig ist immer, dass klar zu erkennen ist, was bloß eine Grafik und was ein Icon mit einer Funktion ist. So zeigt beim Öffnen einer Wetter-App diese direkt eine Grafik an, die über die aktuelle Wettersituation Auskunft gibt.

Abbildung 13.25 Die App »appFigures« zeigt sehr gut, wie man mit Diagrammen in einer App arbeiten kann.

13.2.1 Die Grafik als funktionelle Darstellung

Diese Grafikvariante entspricht genau dem Beispiel der Wetter-App. Die Grafik (Wolke mit und ohne Sonne) zeigt das Resultat, das durch verschiedene Parameter und Funktionen erzeugt wird. Es wird nur die Grafik in verschiedenen Variationen und je nach Szenario angezeigt.

Ebenso gibt es Grafiken, die uns dabei helfen, dynamische Inhalte anzuzeigen. So kann z. B. ein Kuchen- oder Kurvendiagramm so gestaltet sein, dass es eine Grafik ist. Es sind eher »lebende« Grafiken, die sich je nach Einstellungen dynamisch verändern.

Ein weiteres Beispiel für die gezielte, funktionelle Verwendung von Grafiken zeigen On-boarding-Prozesse (siehe Abschnitt 6.3.1, »Das On-Boarding«). Hier werden Grafiken sehr häufig verwendet, die stellenweise sogar animiert sind. Diese Grafiken erklären Hauptfunktionen der App. Das On-Boarding wird teilweise auch als »Betriebsanleitung« verstanden; Grafiken dienen hier als eine vereinfachte Darstellung der Funktionen und beschreiben die Nutzung der App.

Abbildung 13.26 Visuals im On-Boarding der tellly-Hörbuch-App

Grafiken bloß als Schmuckelement?

Grafiken können natürlich auch »langweilige« Views etwas auflockern. Jedoch sollte eine Grafik in solch einem Fall nie das zentrale Element der View sein. Wenn eine View langweilig ist, dann ist die Lösung niemals ein nettes Visual. Verwenden Sie ein Visual als grafisches Element nur dann, wenn der Nutzer einen wirklichen Nutzen davon hat. So kann anstatt einer leeren Liste ein Visual erscheinen, das erklärt, dass man erst etwas hinzufügen muss.

13.2.2 Die Grafik als Maskottchen

Sie kennen sie vielleicht noch: Clippy, die sprechende Büroklammer von Microsoft Word. Vom Ansatz her war Clippy gar keine schlechte Idee – allerdings scheiterte sie an der nervigen Umsetzung. Heute gibt es Maskottchen in weitaus besserer Form.

Ein Beispiel sind die Figuren der Duolingo-App, vor allem die Eule, die auch das Maskottchen der App ist. Während die Grafiken uns dabei helfen, z. B. die französische Sprache zu erlernen, hilft uns die Duolingo-Eule auf eine andere Art und Weise: Sie hat immer einen netten, ermunternden Spruch auf dem Schnabel und erinnert uns, ohne müde zu werden, an das angestrebte Ziel. Sie ist ein Begleiter und unterstützt uns, ohne aufdringlich zu sein.

Abbildung 13.27 Der helfende Vogel in der Duolingo-App

Figuren, die uns in der App unterstützen, an Dinge erinnern oder Mut machen, sind ein gutes Mittel, den Nutzer bei Stange zu halten, und prägen zudem die User Experience der App. Figuren können vor allem dort eingesetzt werden, wo der Nutzer direkt angesprochen werden soll.

Sie können aber gänzlich auf Figuren verzichten. Gestalten Sie die App persönlicher, duzen Sie, anstatt zu siezen. Haben Sie hier und da einen lockeren Spruch parat, das lässt die App persönlicher wirken, ohne dass Sie ein eventuell nerviges Maskottchen haben.

13.2.3 Woher Grafiken beziehen?

Als Designer werden Sie sicherlich selber eigene Grafiken erstellen können, vor allem wenn Sie sich selber eine App überlegen und deren Gestaltung in die eigene Hand nehmen. Wenn Sie mit einem Kunden arbeiten, wird dieser Ihnen aber gegebenenfalls selbst die gewünschten Grafiken bereitstellen oder einen Ansprechpartner nennen, von denen Sie diese erhalten.

Alternativ können Sie solche Illustrationen natürlich auch kaufen, und zwar auf Plattformen wie *www.creativemarket.com* und *www.graphicriver.net*. Jedoch kann es sein, das Sie dort nicht das Richtige finden. Dann haben Sie immer noch zwei Möglichkeiten:

- selber machen
- jemanden beauftragen

Qualitativ hochwertige Grafiken sind häufig teuer und kosten Zeit in der Herstellung. Sie sind meist ein Mittel, das im Gestaltungsprozess recht weit hinten angesiedelt ist, und werden erst spät benötigt. Mein Tipp an Sie: Legen Sie in der Konzeptphase fest,

ob überhaupt Grafiken nötig sind. Falls dies der Fall sein sollte, planen Sie in den Wireframes Ihrer App ein, wo diese Grafiken auftauchen sollen und was der Sinn dieser Grafik sein soll.

Eigener Stil – auch bei Bildern

Kreieren Sie einen eigenen Stil. Verwenden Sie bestimmte Arten von Icons oder bestimmte Filtereffekte für Bilder, um sich von Ihren Konkurrenten abzuheben und eine gezielte visuelle Sprache zu sprechen, und zwar die Sprache, die von Ihrer Zielgruppe oder besser von Ihren Personas gesprochen wird.

13.3 Fotografien

Bilder sagen mehr als tausend Worte. Durch Fotografien haben wir als App-Designer die Möglichkeit, die Aufmerksamkeit des Nutzers zu gewinnen und unsere Nachrichten zu emotionalisieren. Die visuelle Erscheinung eines Objektes auf einem Bild kann im limbischen System des Gehirns einen positiven Reiz hervorrufen. So haben z. B. Fotografien von schönen Landschaften und lachenden Menschen einen positiven Einfluss auf den Betrachter.

Limbisches System

Das limbische System ist ein Teil des menschlichen Gehirns. Es ist eine Funktionseinheit, die sich um die Verarbeitung von Emotionen kümmert und dem Triebverhalten dient.

Jedoch ist der Einfluss von Bildern auf unsere Emotionen nicht das Einzige, was für eine gute User Experience nötig ist.

13.3.1 Die Verwendung von Fotografien in Apps

Der Einsatz von Fotografien auf mobilen Geräten ist dann sinnvoll, wenn sie auch wirklich benötigt werden. In den meisten Fällen hat eine App verschiedene Ziele und verschiedene Zielgruppen. Die Aufmerksamkeit des Nutzers durch eine emotionalisierende Fotografie zu erregen, hat sicherlich einen Vorteil, ist aber mit dem Nachteil verbunden, dass oft User-Interface-Elemente schwerer zu erkennen und zu nutzen sind. Achten Sie bei der Verwendung der Fotografien darauf, dass durch diese nicht die Nutzung und Usability der App leidet und sich die Fotos nicht gestalterisch zu stark in den Vordergrund drängen.

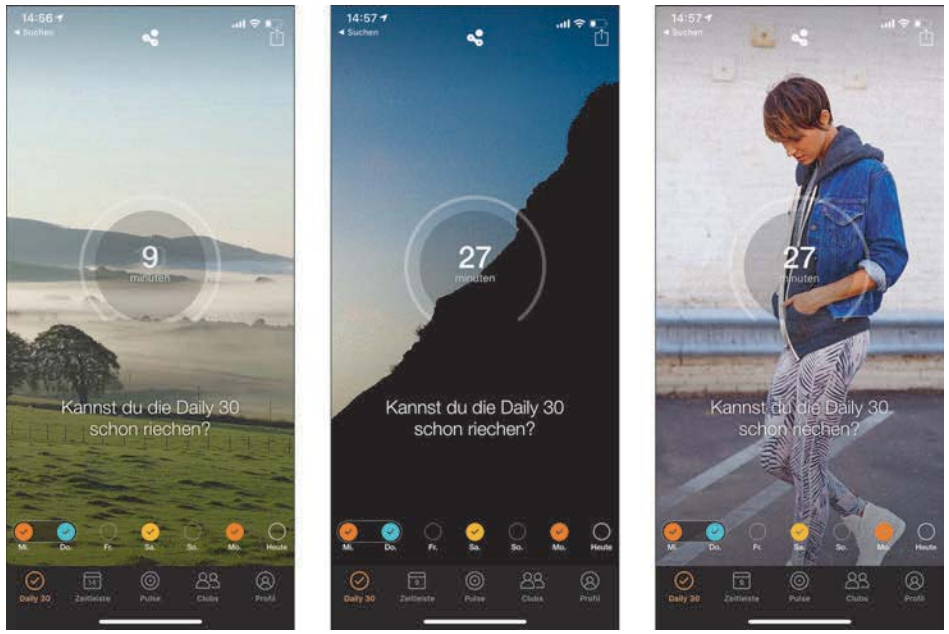

Abbildung 13.28 Hintergrundbilder der Human-App

So lieber nicht

Die Stock-Fotografien können auch richtig danebengehen. Die besten Fehltritte wurden in unterschiedlichen Blogs gesammelt. Viel Spaß!

- *http://shitstockphotos.tumblr.com*
- *http://weird-stock.tumblr.com*
- *www.reddit.com/r/wtfstockphotos*

Ein positives Beispiel für die Verwendung von Fotografien ist die App *Human*. *Human* ist eine Fitnesstracking-App mit verschiedenen Hintergrundbildern. Weitere Hintergrundbilder sind erst dann verfügbar, wenn bestimmte Fitnessziele erreicht wurden. So kann der Einsatz von Bildern nicht nur eine Emotion hervorrufen, sondern dient auch der Motivation, neue Hintergrundbilder freizuschalten. Die App schafft es, ein aufgeräumtes und klar erkennbares User Interface aufzubauen. Die Fotografie drängt sich nicht in den Vordergrund, sondern bleibt dezent im Hintergrund, welcher durch einen leichten Dunkel-zu-Transparent-Verlauf optimiert wurde, um die Icons kontrastreicher darstellen zu können.

Achten Sie beim Einsatz von Fotografien darauf, keine zu »abgegriffenen« Bildmotive zu nutzen (z. B. eine lächelnde Frau mit Headset). Diese Bilder wirken fad und ideenlos. Man spricht auch gerne von »stockig«. Der Begriff leitet sich von den Bilderportalen ab, wo meist der Begriff Stock auftaucht.

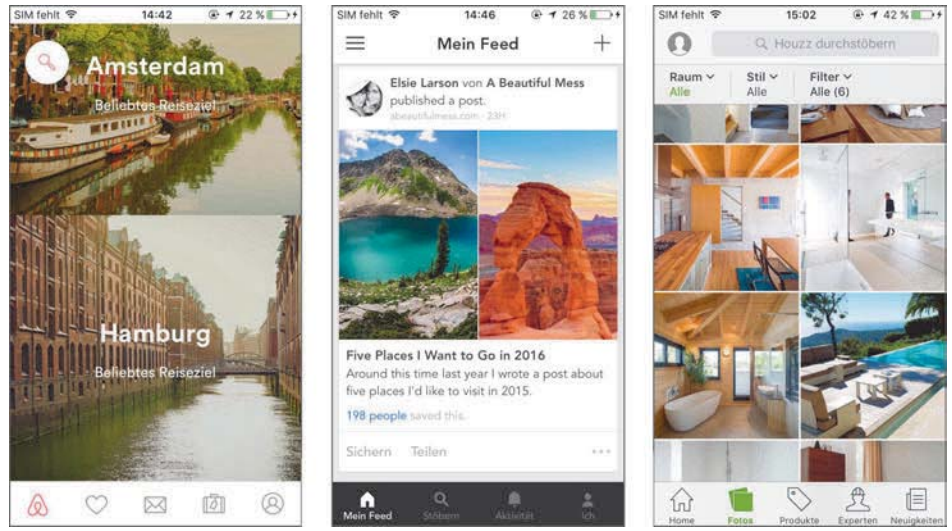

Abbildung 13.29 Bilder wecken mehr Emotionen, als Grafiken oder Icons es können. Von links nach rechts: »Airbnb«, »Bloglovin'«, »Houzz«

13.3.2 Dynamische Bilder

Dynamische Bilder sind Elemente in einem User Interface, die vom Nutzer hinzugefügt werden können. So sind z. B. Facebook-Posts mit Bild ein solcher Fall. Es müssen nicht zwangsläufig Fotografien, sondern können auch Grafiken sein. Als App-Designer haben Sie keinen Einfluss darauf, welche Bilder verwendet werden. Sie haben lediglich die Möglichkeit, die Größe und die Position des Bildes zu bestimmen. In diese Kategorie fallen nicht nur gepostete Bilder, sondern auch Avatare oder Produktplatzierungen.

Abbildung 13.30 Mein Profilbild auf Twitter. Nun haben Sie einen Eindruck von meiner Persönlichkeit, oder?

Die Möglichkeit für den Nutzer, ein individuelles Profilbild zu hinterlegen oder ein Hintergrundbild einzustellen, lässt eine App persönlicher wirken.

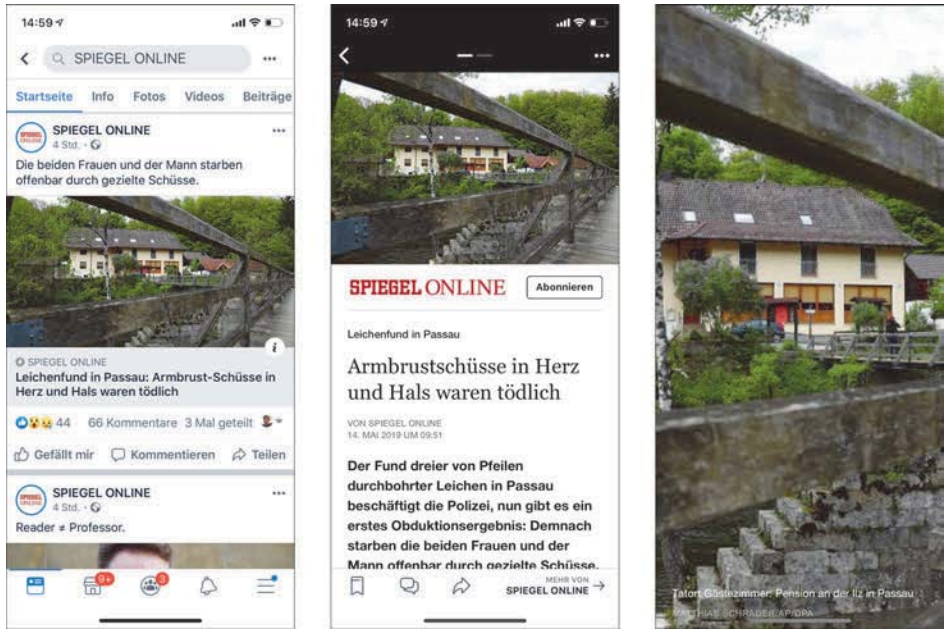

Abbildung 13.31 Dynamische Bilder in der Facebook-App

Das Problem mit dynamischen Bildern ist, dass sie in einem vorher festgelegten Rahmen abgebildet werden müssen, sozusagen einem Platzhalter. Die Applikation befüllt dann diesen Platzhalter mit dem Bild und schneidet dieses je nach genutztem Endgerät zurecht.

Im Grunde sollten Sie bei der Gestaltung darauf achten, dass diese Platzhalterfläche eine Größe hat, die es auf allen Geräten zulässt, den Inhalt des Bildes zu erkennen, ohne dass andere Elemente durch das Bild »verdrängt« werden.

13.3.3 Produktfotografien

Produktbilder sind bei Apps mit Shop-Funktionalität ein wichtiger Punkt. Gerade Fotografien können dazu genutzt werden, App-Nutzer z. B. in eine Kauf- oder auch Kontaktlaune zu versetzen. Die Kaufentscheidung eines Produktes wird maßgeblich von der Fotografie des Produktes geprägt. Der geschickte Einsatz von Produktfotos sorgt im Idealfall dafür, dass das Gehirn stimuliert und ein Gefühl des »Habenwollens« erzeugt wird. In Shops sollten Produktfotos in der Einzelansicht des Produkts so groß wie möglich zu sehen sein.

Verwenden Sie immer das beste und schönste Produktfoto zuerst. Vergessen Sie auch nicht, weitere Produktfotos beizufügen, die Ihr Produkt gut zur Geltung bringen. Bilder tragen stark zur Kaufentscheidung bei; je mehr Details Sie zeigen, umso interessanter ist das Produkt für den Kunden.

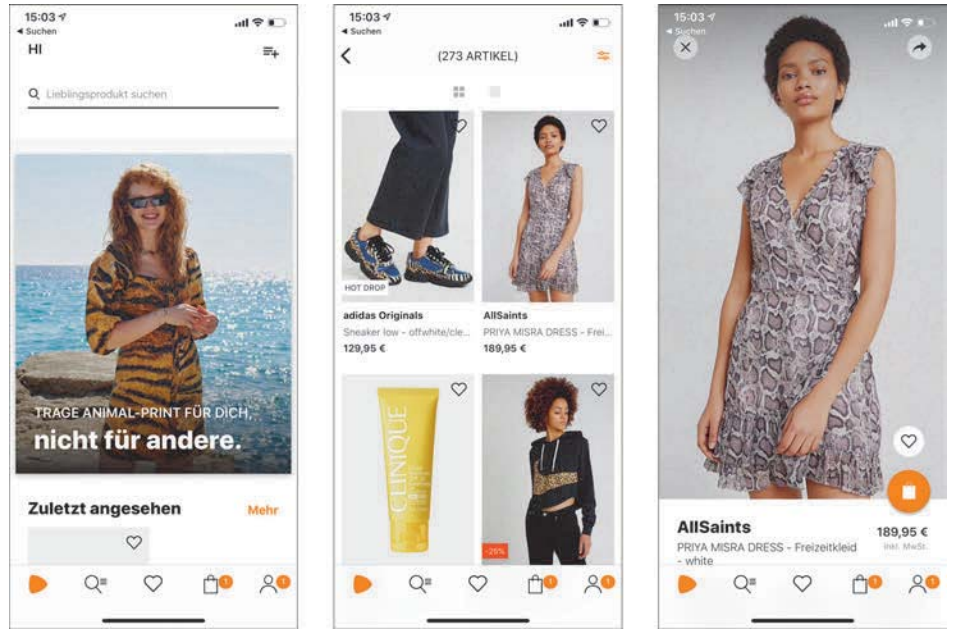

Abbildung 13.32 Schöne Produktfotos in der Zalando-App

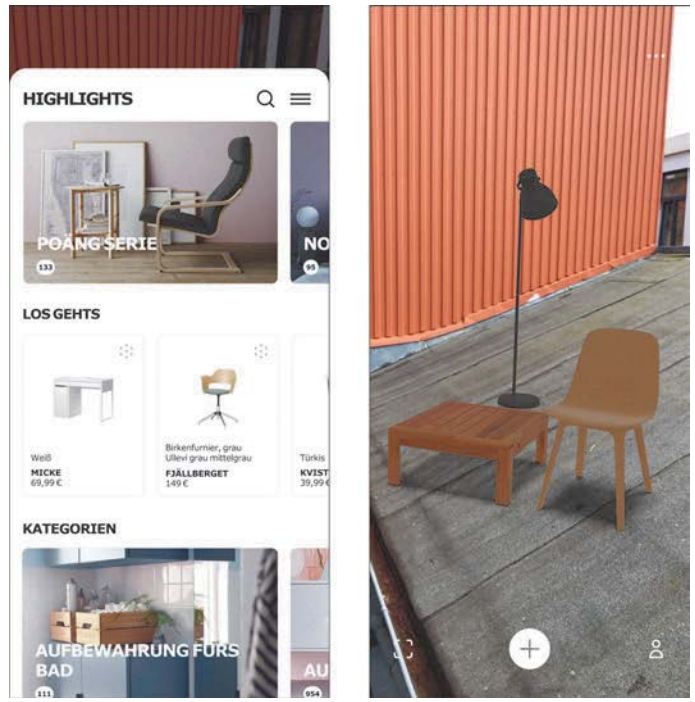

Abbildung 13.33 Mit der IKEA App können Sie Möbel überall in Ihrer Wohnung platzieren, in diesem Fall das Vordach meiner Wohnung.

Sie können aber auch einen Schritt weiter gehen und es wie IKEA machen. Ikea hat bei Veröffentlichung von ARkit von Apple eine App herausgebracht, bei der Sie IKEA-Möbel virtuell in Ihrer Wohnung platzieren können. Das ist sehr beeindruckend und stellt den ersten Schritt für virtuelle Kauferlebnisse dar. Zur iOS-App: *https://itunes.apple.com/de/app/ikea-place/id1279244498?mt=8*.

Apps, die Fotografie und Bildbearbeitung zum Thema haben

Bei solchen Apps ist die Fotografie ein Objekt in einem Prozess, den Sie definieren. Wenn Sie eine derartige App designen wollen, sollten Sie darauf achten, dass die Bildmanipulation kontinuierlich abläuft und der Nutzer sieht, welche Manipulationen er vornimmt. What you see is what you get.

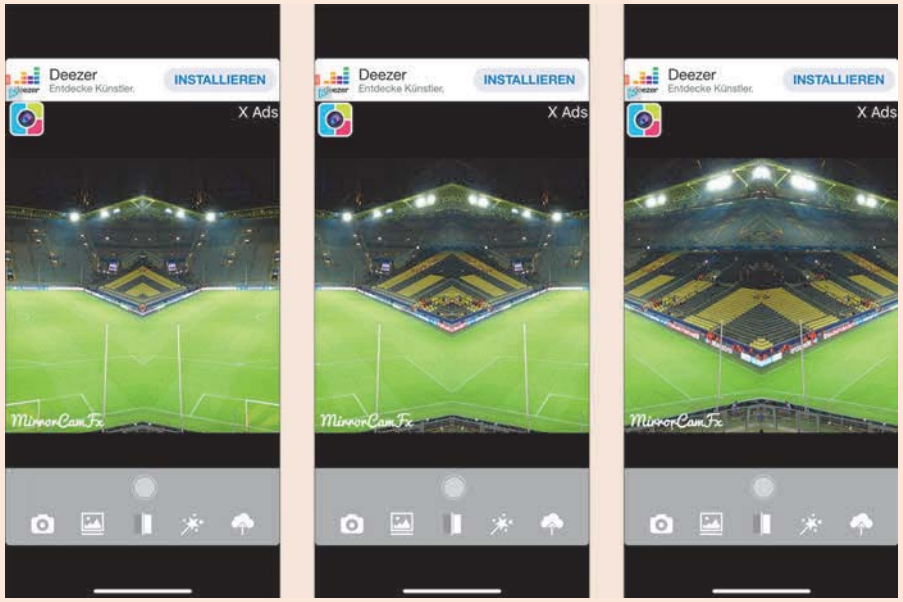

Abbildung 13.34 Bildmanipulationen in der MirrorCamFX App

13.3.4 Woher Fotografien beziehen?

Fotos bekommen Sie aus verschiedenen Quellen. Zum einen gibt es viele Anbieter, die sogenannte *Stock-Fotografien* anbieten, die Sie zum Teil kostenfrei verwenden können. Natürlich können Sie auch jemanden beauftragen, Fotos für Sie zu machen, oder diese selber anfertigen.

Selber machen oder machen lassen

Qualitativ hochwertige Bilder erzeugen auch einen wertigen Eindruck. Dies ist vor allem bei Produktfotografien sehr wichtig. Selbst gemachte Fotos können in der Qualität stark variieren. Wenn Sie leidenschaftlicher Hobbyfotograf sind, ist die Qualität Ihrer Fotos natürlich besser, als wenn Sie keine Kenntnisse haben.

Versuchen Sie, Fotos nur selber zu machen, wenn sie nicht das Hauptmerkmal Ihrer App sind. Als Anmerkung sei gesagt, dass es nicht immer eine teure DSLR-Kamera für Bilder sein muss. Steven Soderbergh, seines Zeichens Hollywood-Regisseur und Macher von Filmen wie »Oceans 11« oder »Erin Brockovich«, drehte 2017 einen Film ausschließlich mit dem iPhone (*https://www.techradar.com/news/steven-soderberghs-new-movie-was-shot-entirely-on-iphones*). Ich will Ihnen damit nur sagen, dass eine teure Kamera nur dann gute Bilder liefert, wenn der Knipser weiß, was er tut.

Für Shopping-Apps empfehle ich, generell professionell erstelltes Bildmaterial zu verwenden. Der Aufwand für professionelle Produktbilder darf nicht unterschätzt werden: Thema finden, Bildaufbau, Bildkonzept, technische Umsetzung, Studio usw.

Kaufen (Fotoagenturen)

Sie können auch Fotos auf diversen sogenannten *Stock-Portalen* kaufen. Hier können Sie in großen Datenbanken nach bestimmten Bildern suchen. Mitunter finden sich qualitativ sehr hochwertige, aber oft auch sehr gestellte, einfache Bilder. Auch sollten Sie ausreichend Zeit einplanen, um das gewünschte Bildmaterial in den Tiefen der Stock-Agentur-Datenbanken zu finden.

Bitte beachten Sie auch die Lizenzbedingungen zu den Bildern; es ist nicht zwangsläufig so, dass jedes Bild auch für alle Medien frei nutzbar ist. Es gibt verschiedene Lizenzmodelle. Jede Plattform hat andere Richtlinien. Folgende Portale gehören zu den größten Anbietern:

- iStock: *www.istockphoto.com*
- Getty Images: *www.gettyimages.com*
- Fotolia: *www.fotolia.com*
- Shutterstock: *www.shutterstock.com*
- Adobe Stock: *http://stock.adobe.com*

Kostenlos (kostenfreie Bilddatenbanken)

Neben den kostenpflichtigen Plattformen gibt es auch Plattformen, die kostenfreie Bilder anbieten. Hierbei sind die meisten Fotografien frei verfügbar und oft auch für den kommerziellen Gebrauch nutzbar. Auch hier sollten Sie die Richtlinien der jeweiligen Plattform beachten. Meist wollen die Fotografen in Ihrer App erwähnt werden. Hierzu können Sie eine Impressumsseite in Ihrer App anlegen und dort den Fotografen erwähnen, ggf. mit Link. Zu den kostenfreien Plattformen zählen unter anderem:

- Flickr: *www.flickr.com*
- Pexels: *www.pexels.com*

- AllTheFreeStock: *http://allthefreestock.com*
- Freeimages: *www.freeimages.com*
- Pixabay: *www.pixabay.com*
- picjumbo: *www.picjumbo.com*
- Unsplash: *www.unsplash.com*

Creative Commons

Creative Commons (CC) ist eine Non-Profit-Organisation, die Lizenzverträge als Hilfestellung für Urheber zur Freigabe von rechtlich geschützten Inhalten anbietet. CC bietet insgesamt sechs verschiedene Standardlizenzen an, die zur Verbreitung kreativer Inhalte wie beispielsweise Fotos, Musikstücke oder Ähnlichem genutzt werden können. Besonders interessant ist der *Creative Commons CC0*: Bilder dieser Lizenzart unterliegen keinem Kopierrecht und können – verändert oder unverändert – kostenlos auch für kommerzielle Anwendungen in digitaler oder gedruckter Form ohne Bildnachweis oder Quellenangabe verwendet werden. Mehr zu Creative Commons: *www.creativecommons.org*.

13.4 Der Export

Haben Sie Grafiken, Icons und Fotografien erstellt und die App so weit vorbereitet, dass sie von einem Entwickler umgesetzt werden kann, benötigt dieser neben einem Styleguide für Farbe und Typografie auch die Icons, Grafiken und Bilder. Diese werden *Assets* genannt.

Assets sind meist Bilddaten im PNG- oder SVG- bzw. PDF-Format. Es sind Dateien, die der Entwickler benötigt, um eine App und ihr grafisches Interface zu entwickeln. Im Grunde baut der Entwickler die Assets zu einem Interface zusammen. Ohne diese Assets gibt es keine Icons, Bilder, Grafiken oder gar Videos.

13.4.1 Styleguide (Gestaltungsrichtlinien)

Ein Styleguide kommt ursprünglich aus dem grafischen Print- und Marketingbereich und beschreibt, wie bestimmte Elemente eines Druckerzeugnisses oder eines digitalen Produkts zu gestalten sind. Styleguides enthalten also generelle Vorgaben für das Design eines Printprodukts oder einer Anwendung. Durch die aufgestellten Richtlinien soll ein einheitliches Erscheinungsbild verschiedener Kommunikationsmittel (Flyer, Broschüre, Website, App etc.) gewährleistet werden, wodurch die Bildung einer Corporate Identity ermöglicht wird.

Folgende Fragestellungen sollen unter anderem mithilfe eines Styleguides beantwortet werden:

- Welche Farben dürfen verwendet werden?
- Welche Schriften werden benutzt? Welche Schriftschnitte?
- Wo wird das Logo positioniert?
- Wie werden Fehler ausgegeben?
- Wie sehen zulässige Interaktionstechniken aus?
- Wie ist das Grid definiert?
- Wie verhält sich die App auf den jeweiligen Endgeräten?
- Was ist erlaubt und was nicht?

Darüber hinaus werden die Zusammenhänge dieser Aspekte erläutert und sogenannte *Dos and Don'ts* gesammelt. Ein solcher Styleguide kann sehr umfangreich dokumentiert sein. Je größer ein Unternehmen, desto umfangreicher ist in der Regel auch der vorliegende Styleguide.

Mit einem Styleguide für eine App verhält es sich ähnlich. Hier werden zusätzlich alle Informationen gesammelt, die ein Entwickler benötigt, um ein Interface umzusetzen. Natürlich sind auch hier Farben, Typografie und das Logo ein wesentlicher Bestandteil des Styleguides. Neben diesen Informationen werden zudem Komponenten und Screens mit geforderter Größe angegeben. So werden Buttons, Eingabefelder, Switches, Navigation, Tab-Bars usw. bemaßt und mit entsprechenden Informationen versehen, die der Entwickler benötigt, um das grafische Interface umzusetzen.

Neben der Bemaßung einzelner Bestandteile der App wie Buttons und Eingabefelder werden auch die Layouts definiert, also Abstände zum bzw. vom Rand des Screens und Positionen der Komponenten.

Gerade wenn mehrere Designer und Entwickler an einer App arbeiten, ist ein Styleguide sehr wichtig. Er ist ein Garant für die Gestaltungskonsistenz und hilft enorm bei der Kommunikation in der Umsetzungsphase.

13.4.2 Assets erstellen

Es gibt einige Dinge zu beachten, wenn Sie Ihre Assets exportieren. Wenn Sie dem Entwickler ohne weitere Erklärung die Quelldatei zuschicken, dürfen Sie nicht erwarten, dass er genau das umsetzt, was Sie sich dabei gedacht haben. Das ist leider ein Wunschtraum, dem viele Designer oft erliegen. Als Designer haben Sie die Verantwortung, dem Entwickler das zu liefern, was er für die Umsetzung benötigt. Nur so garantieren Sie, dass der Entwickler Ihre Ideen auch so umsetzt, wie Sie das geplant haben.

Im Folgenden möchte ich Ihnen zeigen, was Sie beachten müssen und wie Sie Assets in wenigen Augenblicken generieren können. Zu Beginn erläutere ich, welche Einstellungen Sie in den Programmen (hier Adobe XD und Sketch) vornehmen sollten, bevor Sie mit dem Design Ihrer App beginnen.

Anmerkungen für den Export in Adobe XD und Sketch

Vorneweg: Sie müssen sich natürlich nicht sklavisch an die in Abschnitt 13.4, »Der Export«, gezeigten Einstellungen halten.

Ich möchte Sie nochmals darauf hinweisen, dass Sie in jedem Fall ein Layout mit der kleinstmöglichen Auflösung der Plattform kreieren sollten. Sie erinnern sich sicherlich noch an das in Abschnitt 10.4.1, »Die richtige Auflösung und Größe«, beschriebene Skalierungsproblem, wenn mit höheren Auflösungen gearbeitet wird. Das Ziel des Asset-Exports sollte sein, über jede Auflösung hinweg das Optimum an Qualität aus den Assets herauszuholen.

Vektorbasiertes Arbeiten in Adobe XD und Sketch

Wie Sie bereits aus Abschnitt 10.4, »Umsetzung und Design«, wissen, sollten Sie für die Gestaltung einer App nach Möglichkeit immer mit Vektorobjekten arbeiten. Leider ist das nicht immer möglich, vor allem nicht, wenn Sie auch Fotografien in Ihren Designs verwenden. Die Arbeit mit vektorbasierten Dateien erleichtert jedoch vieles. Der größte Vorteil ist, dass sie verlustfrei skalierbar sind. Sketch ist ein vektorbasiertes Program, ebenso Adobe XD und Figma.

Assets und Bounding Boxes

Ein wichtiger Punkt bei der Erstellung der Icon-Assets ist neben der Einheitsgröße (siehe den Abschnitt »Legen Sie eine einheitliche Größe für Icons fest« auf Seite 598) das Bounding der Assets. Man spricht hier auch von *Bounding Boxes*, dem Hüllkörper, in dem sich das Icon befindet. Manche Icons sind schmaler oder niedriger als andere Icons, obwohl diese dieselbe Einheitsgröße haben. Bounding Boxes sorgen dafür, dass ein schmales Icon durch Zugabe von transparenten Bereichen die volle Einheitsgröße erreicht. So erreicht ein Kreis die volle quadratische Einheitsgröße für das Asset, aber ein Oval nicht.

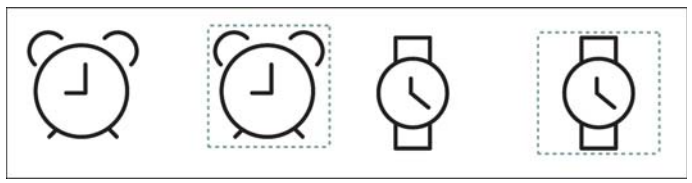

Abbildung 13.35 Links jeweils die Exportgröße ohne Bounding Box und rechts mit Bounding Box (gestrichelte Linie)

Die Exportfunktion von Adobe XD und Sketch wird als Standard immer alle umliegenden transparenten Bereiche des Icons bzw. der Grafik beim Export entfernen. Somit könnte ein schmales Icon mit einer Hüllgröße von 24 × 24 px zu einem Asset mit der Auflösung von 16 × 22 px exportiert werden. Wird nun der Entwickler dieses Icon in der App platzieren, wird es sich nicht harmonisch in andere nebenstehende Icons einreihen. In Adobe XD sowie in Sketch wird die Bounding Box durch die Größe des Artboards bestimmt. Sie benötigen also keine zusätzlichen Masken oder Ähnliches. Einen Hinweis hierzu noch: Deselektieren Sie ggf. Hintergrundflächen, sonst werden diese beim Export mitexportiert.

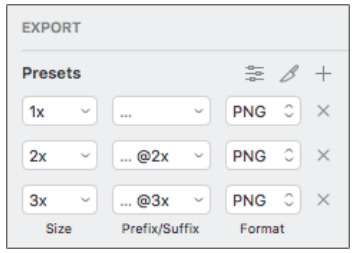

Abbildung 13.36 Die Export-Palette in Sketch

Das Asset Sheet

Jeder Designer hat in der Regel seine eigenen Rezepte, wie er den finalen Schliff an seinen Assets durchführt.

Ich persönlich sammele alle meine Assets und lege sie gesondert ab. In Sketch erstelle ich einfach eine neue Seite im Projekt oder nutze die Symbolseite. Hier reihe ich alle meine zu exportierenden Assets auf. Dabei ordne ich die Elemente in Gruppen an. Alle Grafiken zum On-Boarding werden gruppiert, ebenso User-Interface-Icons usw. Für den Fall, dass ich sehr helle Grafiken verwende, lege ich eine dunkle Hintergrundfläche an, sodass ich die Icons besser erkennen kann. Icons färbe ich immer in Schwarz ein, da Entwickler diese dann später selbstständig durch deren Programmierung einfärben können.

Ich gruppiere alle Icons und Grafiken nach Größe und benenne die Artboards mit dem Namen des Icons. Ich versuche hierbei, einer Systematik zu folgen, und nutze ein Präfix für jede Art von Asset: Für Icons verwende ich »ic_«, für Visuals »vi_« und für Bilder »img_«. Das hat den Vorteil, dass der Entwickler erkennt, um welche Art Asset es sich handelt. Zudem wird bei Android die Voranstellung »ic_« verlangt.

Verzichten Sie auf Umlaute wie ä, ö, ü, da auch sie beim Export zu Problemen führen können. Bei Android-Assets sollten Sie keine Bindestriche verwenden, denn diese können zu Fehlern führen, nutzen Sie stets den Unterstrich _.

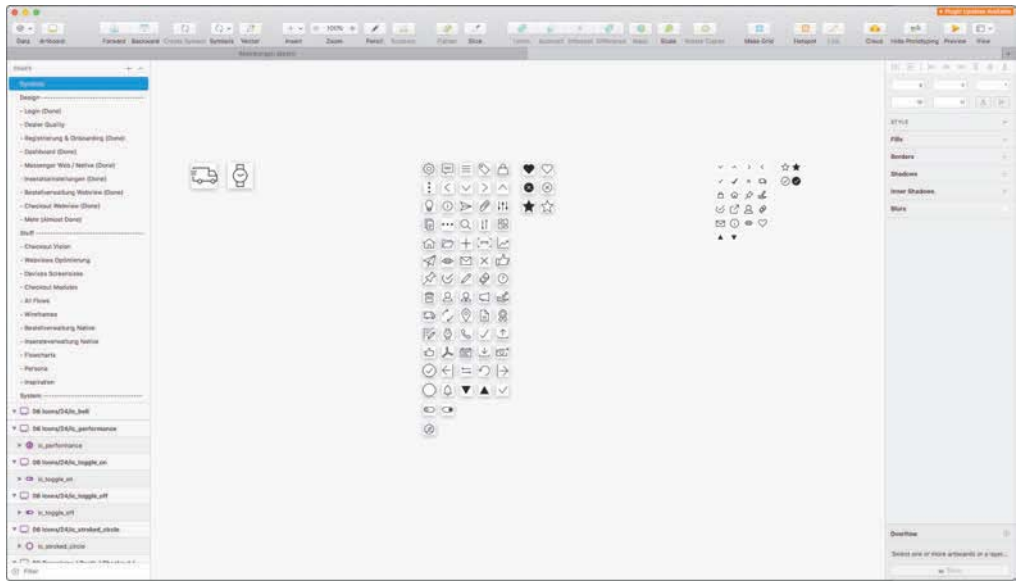

Abbildung 13.37 Symbolseite in Sketch mit vielen Icons bereit zum Einsatz

Im nächsten Schritt kontrolliere ich, ob alle Elemente die entsprechenden Einheitsgrößen haben, falls nötig, nehme ich Optimierungen an den Größen vor. Icons positioniere ich immer mittig, Grafiken je nach Ausrichtung.

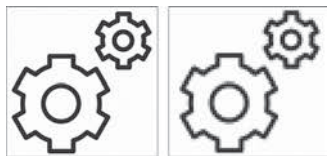

Abbildung 13.38 Rechts ist der Pixelmodus in Sketch inaktiv und links aktiv.

Wenn ich mit Sketch arbeite, schaue ich mir jedes Icon im Pixelmodus an. Nun optimiere ich die Icons und sorge dafür, dass sie pixelperfekt positioniert sind, denn im Laufe eines Designprozesses passiert es gern mal, dass einige Icons versehentlich skaliert oder verschoben wurden und nicht mehr im Pixelraster liegen. Wie bereits erwähnt, werden in Sketch Vektoren basierend auf einem Pixelraster ausgerichtet. Durch Skalieren oder Verschieben von Elementen kann es vorkommen, dass Icons nicht mehr an diesem Pixelraster ausgerichtet werden. Das resultiert in einem schlechtem Anti-Aliasing der Grafik, was dazu führt, dass Icons oder Grafiken, wenn sie exportiert werden, nicht mehr scharf dargestellt werden. Hierbei kontrolliere ich ebenso die Farbe sowie im Fall von Stroked Icons die Strichstärke. Nun kann der Export beginnen.

13.4.3 Der Export unter Sketch

Für den Export in Sketch markiere ich zunächst alle zu exportierenden Artboards. Durch ein Klick auf das »Make Exportable« im unteren rechten Bereich werden alle markierten Objekte für den Export definiert. Anschließend können Sie auswählen, ob Sie für iOS oder Android exportieren möchten. Die Auswahl versteckt sich hinter dem Filtersymbol. Sie können in Sketch auch eigene Vorlagen erstellen oder gänzlich andere Formate und Größen exportieren.

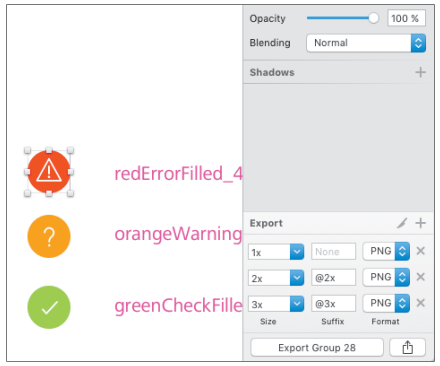

Abbildung 13.39 Unten rechts finden Sie die Exportfunktion in Sketch.

Von Haus aus exportiert Sketch Assets für iOS und Android. Sie müssen nur noch den Export-Layers-Button betätigen, und der Export ist erledigt.

Unter iOS werden die unterschiedlichen Geräteauflösungen durch den Suffix (*@2x*) bestimmt, in Android sind es Ordner. Sketch kümmert sich automatisch, je nach Auswahl, um den korrekten Export. Bei einer App mit wenigen Assets ist das noch recht schnell bewerkstelligt. Bei Apps mit sehr vielen Assets dauert das sehr lange. Aber es gibt einen eleganteren Weg. Später dazu mehr.

13.4.4 Der Export unter Adobe XD

Wie unter Sketch platziere ich die Icons für den Export in Adobe XD alle einzeln auf Artboards und gruppiere diese. Im Grunde gehe ich genauso vor wie in Sketch. Um nun die Assets zu exportieren, wähle ich alle Artboards zum Export aus und öffne den Dialog: Datei • Export • Markierte. Anschließend folgt ein Dialog, bei dem ich bestimmen kann, welches Dateiformat ich wünsche und für welche Plattform ich exportieren möchte. Zudem kann ich die Grundgröße meines Layouts bestimmen. Das ist sehr wichtig, da Adobe XD je nach Ausgangsgröße die Assets angepasst exportieren kann.

Ich glaube, Sie können sich vorstellen, wie aufwendig das Ganze bei einem Umfang von rund 50 Assets wird. Es gibt aber tolle Tools, die den Designer beim sogenannten

Hand-off unterstützen. Eines davon ist das Tool *Zeplin* (*https://zeplin.io*), das ich etwas später noch vorstelle.

13.4.5 Was geht nach dem Export an den Entwickler?

Natürlich braucht der Entwickler Assets. Unter iOS reicht ein Ordner mit allen Assets. Unter Android braucht der Entwickler alle *drawable*-Ordner. Aus Erfahrung heraus kann ich sagen, dass es nie eine einzige, finale Übergabe der Assets gibt. Es gibt immer Nachbesserungen, oder neue Assets kommen hinzu. Einmal ausgeliefert und integriert, können Sie im Teststadium die App testen und den Einsatz Ihrer Assets kontrollieren. Nutzen Sie hier die Möglichkeit, die Assets zu optimieren und zu justieren. Weisen Sie den Entwickler im nächsten Turnus darauf hin, dass er fehlerhafte Assets durch optimierte austauschen soll.

Neben diesen Assets benötigt der Entwickler weitere Assets, die gestaltet werden müssen und die nicht zwangsläufig Teil des Interfaces sind. Dazu gehören das *App-Icon* und das *Launch-Image* der Applikation.

Das App-Icon wird in verschiedenen Größen benötigt. Im Internet gibt es viele Tools, die Ihnen helfen, App-Icons und Launch-Images recht schnell zu erstellen. Ebenso finden Sie Websites, die Ihnen auf Basis eines 1.024 × 1.024 px großen App-Icons die benötigten Größen erstellen. Zwei Tools möchte ich Ihnen hier kurz nennen:

- *Android Asset Studio*: Das Android Asset Studio ist eine Webseite, auf der Sie verschiedene Assets – sei es das Launch-Image, das App-Icon oder sogar Assets für den Entwickler – aufbereiten können. Mehr Infos unter: *https://romannurik.github.io/AndroidAssetStudio/index.html*.
- *App Icon Template*: Unter App Icon Template finden Sie eine Photoshop-Vorlage inklusive Actions, die Sie nutzen können, um Ihr App-Icon für Android zu exportieren. Hier gestalten Sie einfach Ihr App-Icon in dem Photoshop-Dokument, führen anschließend die Aktion aus und exportieren somit die benötigten App-Icons. Mehr Infos unter: *http://appicontemplate.com/android/*.

> **App-Icon gestalten**
>
> Auf die Besonderheiten der Gestaltung des App-Icons gehe ich noch ausführlich in Abschnitt 14.3, »Warum das App-Icon so wichtig ist«, ein.

Größe App-Icon (Android)

Für Android benötigen Sie die folgenden Größen für das Standard App-Icon und Launch-Image:

Verwendung	Größe
MDPI	48 × 48 px
HDPI	72 × 72 px
XHDPI	96 × 96 px
XXHDPI	144 × 144 px
XXXHDPI	192 × 192 px
Playstore App Icon	512 × 512 px

Tabelle 13.1 Größen der App-Icons für Android (rechte Tabelle)

Adaptive Icons/Symbol für Android

Ab Android 8.0 können Entwickler sogenannte Adaptive Icons unter Android nutzen. Durch die enge Auflistung auf dem App Drawer schaffen die App-Icons Unruhe. Um das zu ändern, entwickelte Google das Adaptive App-Icon.

Im Grunde besteht das App-Icon aus zwei Elementen: dem Vorder- und dem Hintergrund. Der Vordergrund beinhaltet das App-Icon, bzw. das Logo oder Signet der App, der Hintergrund besteht aus einer neutralen Grafik, welche die Freifläche entsprechend auffüllt. Das Betriebssystem legt beide dann übereinander, und somit kann das Icon sehr einfach in verschiedene Formen gebracht werden.

Durch die Trennung des Vorder- und des Hintergrunds können Animationseffekte in die Icons gebracht werden. Somit kann das App-Icon in verschiedenen Bereichen unter Android eingesetzt werden, z. B. im Teilen-Overlay oder in den Einstellungen. Hier erfahren Sie mehr über Adaptive Icons: *https://developer.android.com/guide/practices/ui_guidelines/icon_design_adaptive*.

Was Sie bei der Erstellung eines Adaptive Icons wissen müssen:

- Vorder- und Hintergrund haben eine Größe von: 108 × 108 dp.
- Der innere Teil (72 × 72 dp) des Icons erscheint im maskierten Viewport.
- Das Betriebssystem reserviert den äußeren Bereich mit 36 dp an allen vier Seiten, um interessante Animationen wie den Parallaxen-Effekt oder Pulsieren zu nutzen.

Größe Launch-Image (Android)

Verwendung	Größe
MDPI	480 × 320 px
HDPI	800 × 480 px

Tabelle 13.2 Größen der Launch-Images unter Android (linke Tabelle)

Verwendung	Größe
XHDPI	1280 × 720 px
XXHDPI	1600 × 960 px
XXXHDPI	1920 × 1280 px

Tabelle 13.2 Größen der Launch-Images unter Android (linke Tabelle) (Forts.)

Größe von App-Icon (iOS)

Neben dem App-Icon für den Homescreen benötigt iOS weitere Icons für weitere Bereiche innerhalb des Betriebssystems, z. B. in der Spotlight-Suche in iOS.

Verwendung	Größe
iPhone	180 × 180 px (60 × 60 pt @3x) 120 × 120 px (60 × 60 pt @2x)
iPad Pro	167 × 167 px (83.5 × 83.5 pt @2x)
iPad, iPad Mini	152 × 152 px (76 × 76 pt @2x)
iTunes App Store Artworks (das App-Icon, das im App Store angezeigt wird)	1024 × 1024 px (1024 × 1024 pt @1x)

Tabelle 13.3 Benötigte Größen des App-Icons bei Apple

Größe Launch-Image (Apple)

Wird eine App gestartet, wird, kurz bevor die App wirklich geladen ist, ein Bild angezeigt, das sogenannte *Launch-Image*, auch *Splashscreen* genannt.

Abbildung 13.40 Splashscreen von »Human App«, »Lifesum«, »Eventbrite«, »Kickstarter«

Verwendung	Größe
iPhone 4, 4S	640 × 960 px
iPhone 5, 5S, 5C	640 × 1136 px
iPhone 6s	750 × 1334 px
iPhone 6s+	1242 × 2208 px
iPhone 7	750 × 1334 px
iPhone 7+	1242 × 2208 px
iPhone 8	750 × 1334 px
iPhone 8+	1242 × 2208 px
iPhone X	1125 × 2436 px
iPhone XR	828 × 1792 px
iPhone XS	1125 × 2436 px
iPhone XS Max	1242 × 2688 px
iPad Mini 4	1536 × 2048 px
iPad 9.7"	1536 × 2048 px
iPad Pro 10.5"	1668 × 2224 px
iPad Pro 11"	1668 × 2388 px
iPad Pro 12.9"	2048 × 2732 px

Tabelle 13.4 Benötigte Launch-Images-Größe für eine Apple-App

13.5 Tools für den Export und Styleguides

Im letzten Teil des Kapitels möchte ich Ihnen noch einige Tools vorstellen, die Ihnen die Arbeit und auch die Übergabe an den Entwickler etwas erleichtern.

13.5.1 Tools & Plugins für Sketch über die »Sketch Toolbox« installieren

Sketchpacks ist ein Sketch-Plugin für Mac, das alle verfügbaren Sketch-Plugins von GitHub sammelt und sich für Sketch installieren lässt. Die Applikation arbeitet unab-

hängig von Sketch und installiert Plugins für Sketch. Das Programm finden Sie hier: *https://sketchpacks.com*. Hierüber lassen sich etwa folgende nützliche Plugins installieren:

- **Marketch:** Dieses Plugin erstellt einen komplett bedienbaren Styleguide. Hier können Maße, Farben, Größen und Positionen abgelesen werden. Diese Informationen lassen sich dabei ganz einfach über den Webbrowser abrufen. Das ist sehr nützlich, da man hier Abstände und Größen ablesen kann.

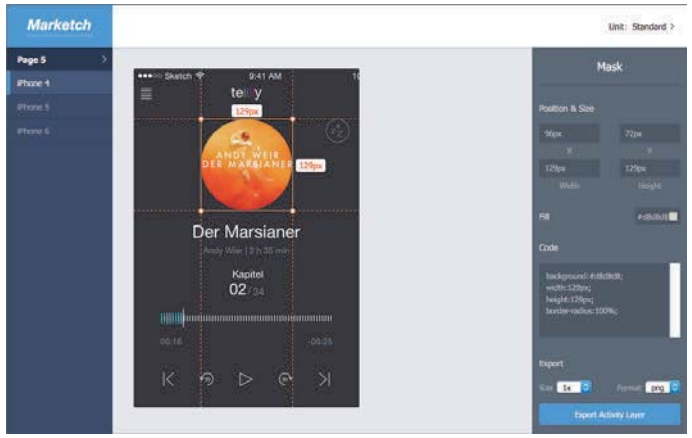

Abbildung 13.41 Mit dem »Marketch«-Plugin lassen sich Styleguide-Informationen bequem über den Webbrowser auslesen.

- **Rename IT:** Ein Renaming-Plugin, welches mir ungemein dabei hilft, eine große Anzahl an Artboards oder Objekten umzubenennen. Sie wollen doch nicht gewisse Prä- und Suffixe ständig von Hand ändern, oder? *Rename It* ist verfügbar für Sketch, aber auch für Adobe XD.

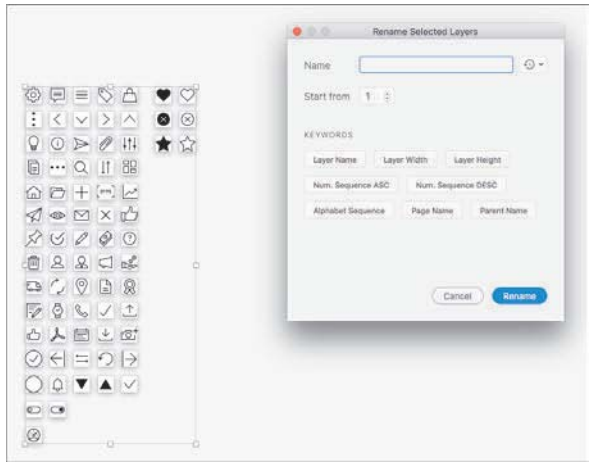

Abbildung 13.42 Rename it-Plugin für Sketch

- **Zeplin oder Sympli:** Beides sind sogenannte Hand-off-Tools und unterstützen den Designer, aber auch den Entwickler dabei, die Designs zu inspizieren. Hierbei werden die erstellten Assets über ein Plugin in Sketch auf die Zeplin-Cloud-Lösung hochgeladen und analysiert. Der Clou? Sie müssen lediglich die Artboards einmal zum Export markieren, vergessen Sie die Einstellungen für iOS oder Android. Zeplin übernimmt und kümmert sich um alles. Ebenso können Farben und Schriften festgelegt werden und mit Variablen Namen der Entwickler benannt werden. Der Entwickler hat nun die Möglichkeit, die Abmessungen zu inspizieren und Assets zu exportieren. Zudem gibt es die Möglichkeit, Kommentare zu hinterlegen.

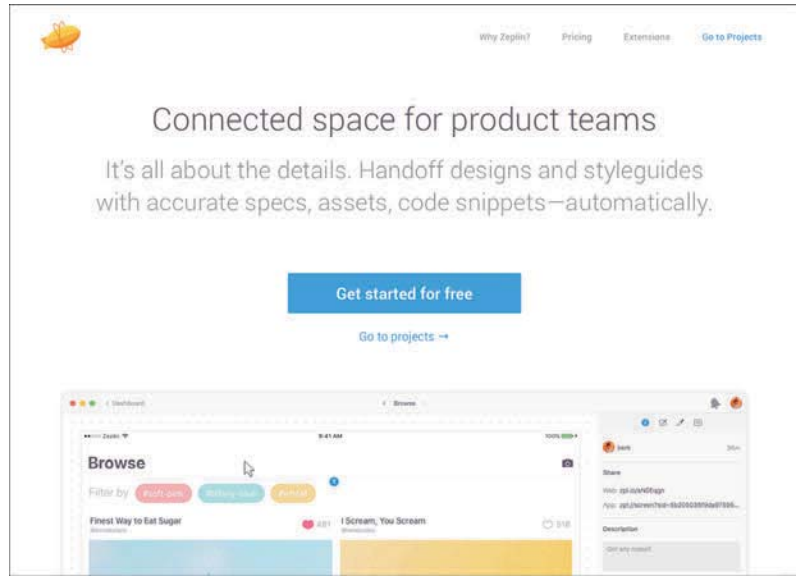

Abbildung 13.43 Zeplin-Webseite

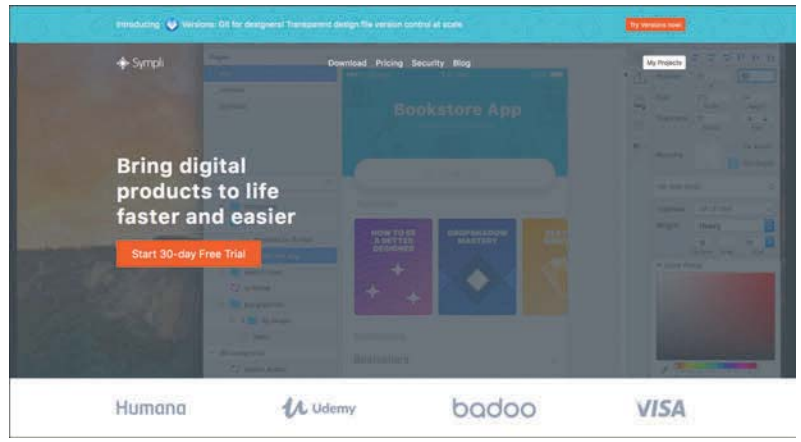

Abbildung 13.44 Sympli-Webseite

13.5.2 Plugins für Adobe XD

Seit Ende 2018 gibt es auch endlich Plugins für Adobe XD. Somit gibt es noch nicht so viele Anbieter und deshalb auch noch nicht sonderlich viele Plugins. Hier finden Sie die aktuellsten: *www.adobe.com/de/products/xd/resources.html*.

Die meisten beschränken sich auf die Erstellung von Grafiken oder andere kleine Helfer. Das Plugin Rename It gibt es bereits und ein Plugin für das Zeplin-Tool. Sicherlich werden wir in den kommenden Jahren weitere Plugins sehen, die uns beim Export unterstützen.

Nach diesem Kapitel sind Sie über alle wichtigen Aspekte der App-Gestaltung informiert und können souverän mit Usability, User Experience, Farbe, Typografie und Bildern jonglieren, ohne dabei den Nutzer Ihrer App aus den Augen zu verlieren. Und auch um den Export Ihrer Assets haben Sie sich schon gekümmert. Jetzt muss die App noch in die App Stores. Hier ist vor allem die ansprechende Präsentation der App, sprich das richtige Marketing, wichtig. Dieses Thema soll den Abschluss dieses Buchs bilden.