Vielen Dank, dass Sie sich für ein eBook von entwickler.press entschieden haben!

Mehr eBooks und alle Informationen zu unserem Verlagsprogramm finden Sie unter:

www.entwickler-press.de

Viel Spaß,
Ihr entwickler.press-Team

zum Inhaltsverzeichnis

zum Index

Kapitel 1

Mit den Pfeiltasten können Sie durch das Dokument navigieren, mit der ESC-Taste beenden Sie den Vollbildmodus.

Tobias Bosch, Stefan Scheidt, Torsten Winterberg

Mobile Web-Apps mit JavaScript

Leitfaden für die professionelle Entwicklung

entwickler.press

Tobias Bosch, Stefan Scheidt, Torsten Winterberg
Mobile Web-Apps mit JavaScript
ISBN: 978-3-86802-273-5

Ein Imprint der Software & Support Media GmbH

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

Ihr Kontakt zum Verlag und Lektorat:
Software & Support Media GmbH
entwickler.press
Darmstädter Landstraße 108
60598 Frankfurt am Main
Tel.: +49 (0)69 630089-0
Fax: +49 (0)69 630089-89
lektorat@entwickler-press.de
http://www.entwickler-press.de

Lektorat: Sebastian Burkart
Korrektorat: Lisa Pychlau
Satz: Laura Acker

Inhaltsverzeichnis

Danksagung

Wir bedanken uns herzlich bei der Firma OPITZ CONSULTING GmbH, ohne deren Unterstützung dieses Buch nicht zustande gekommen wäre. Bei unseren Familien möchten wir uns dafür bedanken, dass wir so manches Mal der Arbeit an dem Buch den Vorrang vor unserem Familienleben gegeben haben. Für ihre inhaltlichen Beiträge zum Buch bedanken wir uns bei Herrn Farid Zaid und Herrn Jörg Bredlau. Frau Jennifer Zimmermann danken wir für die Erstellung der wunderbaren Abbildungen. Ein persönlicher Dank von Stefan Scheidt geht an Herrn Dr. Dirk Koller für die Inspiration durch sein Buch „iPhone-Apps entwickeln“. Zum Schluss bedanken wir uns herzlich bei entwickler.press für die Unterstützung bei der Erstellung und Veröffentlichung des Buchs.

Das Autorenteam Tobias Bosch, Stefan Scheidt
und Torsten Winterberg

1 Einleitung

1.1 Warum dieses Buch?

Sie nehmen dieses Buch in die Hand und fragen sich vielleicht, was das Besondere an diesem Werk sein mag. Warum macht sich jemand die Mühe, eine weitere Publikation zur langen Liste der schon existierenden JavaScript-Bücher hinzuzufügen? Nun, die Antwort ist einfach: Wir hatten gar nicht vor, ein Buch zu schreiben. Als wir damals begonnen haben, uns mit der Entwicklung von mobilen Webanwendungen zu beschäftigen, haben wir viel Recherche betrieben, die uns die Einarbeitung leichter machen sollte. Dabei hat sich jedoch herausgestellt, dass wir scheinbar eine seltsame Zielgruppe darstellen, die (bisher) nicht gut in der Literatur adressiert wird. Wir haben unseren Hintergrund in der Entwicklung von Enterprise-Java-Anwendungen und sind es gewohnt, kleinere bis sehr große Projekte professionell abzuwickeln. Dabei werden wir unterstützt von agilen, flexiblen Softwareentwicklungsprozessen, wir entwickeln im Geiste des Software-Craftmenship und legen sehr viel Wert auf testgetriebene Ansätze. In der Literatur haben wir viel Hilfe gefunden in Form von Büchern, die Ausschnitte beleuchten, etwa neue HTML5-Features, AJAX-Backend-Kommunikation, Sprachfeatures von JavaScript oder Zusammenfassungen sinnvoller Entwurfsmuster. Unser Wunsch nach einer kompletten, durchgehenden Anleitung, wie man professionell auf hohem Niveau nun Webanwendungen schreiben kann, die auf mobilen Endgeräten laufen, ging nicht in Erfüllung.

Also haben wir uns dem Thema ohne eine solche Anleitung genähert, was teilweise recht schmerzhaft war. Daher beschlossen wir Mitte 2011, unsere gewonnenen Erkenntnisse in Form eines Workshops weiterzugeben. Diesen Workshop mit dem Titel „Mobile JavaScript-Web-Apps professionell entwickeln“ hielten wir zum ersten Mal auf der W-JAX-

Konferenz in München im November 2011 und dann diverse weitere Male, z. B. auf der MobileTech Conference Spring 2012 und auch der JAX 2012 mit sehr gutem Feedback.

Dieses Buch basiert nun auf dieser Workshopserie und liefert den Leitfaden zur professionellen Entwicklung mobiler JavaScript-Web-Apps, den wir lange Zeit selbst so schmerzlich vermisst haben.

1.2 Warum mobile Webapplikationen?

Der wirkliche Hype um mobile Lösungen wurde wohl unstrittig ca. 2008 durch die Markteinführung von Apples iPhone ausgelöst. Ausgestattet mit einem exzellenten App-Konzept konnten viele spannende Anwendungen und Funktionalitäten einfach nachgerüstet werden. Das Konzept des App Store begann sich zu etablieren. Zwei Jahre später drängten dann die Android-basierten Smartphones auf den Markt, einhergehend mit einer großen Vielfalt an unterschiedlichen Geräten und Displaygrößen, die heute abgerundet werden von einem Angebot an Tablet-PCs, ebenfalls mit sehr unterschiedlichen Auflösungen und Formfaktoren. Wie groß die Rolle von ehemals etablierten Plattformen wie RIM/BlackBerry und Microsoft/Nokia sein wird, lässt sich heute nur spekulieren. Diese Situation macht die Erstellung von nativen Apps mit einer breiten Plattformunterstützung sehr aufwändig, da Code nicht einfach wiederverwendet werden kann. Jede Plattform bringt ihre eigenen Sprachen, Entwicklungswerkzeuge und Eigenarten mit.

Was kann man also tun, um den Entwicklungsaufwand einzugrenzen? Für die Frage, welche Plattformen gerade „hip“ sind und in jedem Fall bedient werden wollen, lassen sich zahlreiche Statistiken der einschlägigen Analysten auftreiben, die sich mehr oder weniger deutlich voneinander unterscheiden. Anfang 2012 werden hier typischerweise iOS und Android genannt. Kann man die Plattformbreite derart einschränken, dann muss man sich lediglich darum kümmern, die nativen Apps für

diese beiden Plattformen mit ihren unterschiedlichen Formfaktoren zu erstellen. Dies erscheint vom Aufwand her tragbar.

Hat man aber eine größere Gerätevielfalt abzudecken, kommt schnell der Wunsch nach einer Cross-Plattform-Technologie auf, um eine App nur einmal entwickeln und dann mit keinen oder nur minimalen Anpassungen überall verwenden zu können. Hierzu bieten sich natürlich die Browser an und das mittlerweile allgegenwärtige HTML5, das antritt, das Cross-Plattform-Versprechen einzulösen.

Vielerorts wird nun behauptet, dass mobile Web-Apps mit HTML5 von nativen Anwendungen nicht zu unterscheiden seien und daher immer gleichwertig zu den nativen Apps zu betrachten seien. Aus unserer Sicht ist dies keineswegs der Fall. Native Apps sind, was das Bedienungs-Feeling angeht, immer „sexier" als die Pendants mit HTML5. Wenn der Erfolg einer App davon abhängt, sich gegen viele gleichartige Apps durchzusetzen und möglichst oft in einem App Store heruntergeladen zu werden, meist in typischen B2C-Szenarien, wird man um eine native Implementierung nicht herumkommen. Für viele Einsatzszenarien sind die HTML5-Web-Apps aber schlicht gut genug, beispielsweise dann, wenn es für die umzusetzende Aufgabe keine alternative App geben kann, wenn es nicht darauf ankommt, dass sich die App 100 % nativ und „sexy" anfühlt, typischerweise dann eher im B2B-Umfeld. Hier kann sich dann der Cross-Plattform-Vorteil durchsetzen und eine möglichst große Bandbreite an Plattformen (z. B. iOS, Android, Windows Phone und BlackBerry) abgedeckt werden.

Nicht zu unterschätzen ist auch der Vorteil des HTML-Ansatzes, wenn es um schnelle Updates der Anwendung geht. Da der Code jeweils vom Server neu geladen wird, kommen Änderungen sofort zum Tragen, wohingegen Apps im App Store mitunter aufgrund der komplexen Freigabeprozesse längere Updatezyklen haben. Selbstverständlich sind hier auch Mittelwege denkbar, bei denen native Apps HTML-Code nachladen und diesen in dynamischeren Bereichen sinnvoll anzeigen (Stichwort Hybrid-Apps).

Ein weiterer guter Grund für die Entscheidung für die Web-App liegt darin begründet, dass wir einen starken Hintergrund in der Enterprise-Java-Welt haben, uns in der Implementierung klassischer Webanwendungen sehr gut auskennen und unsere Kenntnisse sehr schnell auf die Entwicklung mobiler Webanwendungen ausdehnen können.

1.3 Warum JavaScript?

JavaScript erlebt gerade einen regelrechten Aufschwung. In der Vergangenheit wurde die Sprache etwas belächelt, weil sie im Wesentlichen zum „Aufhübschen" von Webseiten verwendet, aber nicht als „echte" Sprache für professionelle Anwendungsentwicklung gesehen wurde. Mit der enormen Ausbreitungsgeschwindigkeit von HTML5 dreht sich dieser Trend: JavaScript ist auf dem Weg, sich als Sprache zur Anwendungsentwicklung und Oberflächenprogrammierung zu etablieren. Man kann durchaus behaupten, dass JavaScript für die Webentwicklung das ist, was Java für Enterprise Computing im Backend bedeutet. JavaScript enthält viele sehr moderne Eigenschaften für eine 15 Jahre alte Programmiersprache. Wenn man die Sprache richtig einsetzt, kann sie sehr gut mit aktuellen Vertretern wie Groovy oder Ruby mithalten. Sie hat als dynamisch typisierte, objektorientierte Skriptsprache mit prototypbasierter Vererbung viele für den Java-gewohnten Entwickler ungewohnte Eigenschaften. Es braucht seine Zeit, sich daran zu gewöhnen und zu lernen, idiomatisch gutes JavaScript zu schreiben. Je früher man sich damit auseinander setzt, desto besser. Auch auf dem Server hält JavaScript mit Laufzeitumgebungen wie Node.js Einzug und platziert sich dort als sehr valide Lösung für bestimmte Problemkategorien, beispielsweise Anwendungen, die eine große Anzahl konkurrierender Zugriffe auf „langsame Ressourcen" unterstützen müssen. Es gibt zwar zunehmend mehr Werkzeuge, die das Programmieren in JavaScript vor dem Entwickler verbergen, wie z. B. Google Web Toolkit, CoffeeScript, ClojureScript oder auch neue Sprachansätze, wie Google Dart. Dennoch ist eine fundierte Kenntnis der Grundlagen unentbehrlich.

1.4 Single Page Web-Apps

Die innere Architektur der Web-Apps ändert sich stark. Man entfernt sich von dem gängigen Muster klassischer Webanwendungen, die dadurch geprägt sind, dass die Seitennavigation in der Regel durch ein MVC-Muster umgesetzt wird, was den Pageflow Controller auf dem Server ansiedelt. Die Interaktion mit einer Webseite löst einen Request gegen den Webserver aus, hier ermittelt der Pageflow Controller, welche Seite als Nächstes angezeigt werden soll und der Server schickt diese Seite als Response an den Browser zurück. Bei Web-Apps hingegen wird der Client intelligenter. Man verlagert den Pageflow Controller in der Regel auf den Client, meist in JavaScript realisiert. Hier spricht man dann von den so genannten Single Page Web-Apps. Dies liegt darin begründet, dass beim einmaligen Laden einer HTML-Seite gleich die ganze Web-App geladen wird, also eine einzige HTML-Seite, von der aber nur ein Teil angezeigt wird. Soll eine andere Seite angezeigt werden, so blendet der Pageflow Controller auf Clientseite lediglich einen anderen Bereich der schon geladenen Seite ein. Auf diese Weise wird der Request-Response-Zyklus zum Server und zurück vermieden. Dieser tritt lediglich beim Aufruf von Services auf, die per SOAP oder aber meist REST angesprochen werden. Hier leistet beispielsweise ein Enterprise Service Bus (ESB) gute Dienste bei der Protokollkonvertierung, wenn das bestehende Service-Backend nur SOAP spricht, im Frontend aber REST zum Einsatz kommen soll.

Im Ergebnis handelt es sich bei Single Page Web-Apps also mehr um Clientapplikationen im Sinne der Client-Server-Programmierung als um Webapplikationen im Sinne von serverseitig generierten Webseiten. Mobile Webapplikationen, die auf diese Weise entwickelt wurden, integrieren sich hervorragend in Gesamtarchitekturen, bei denen das Backend über ein RESTful API angesprochen werden kann. Optimal für das Zusammenspiel ist eine Kommunikation auf Basis der JavaScript Object Notation (JSON). Weitere Vorteile dieser Applikationsarchitektur sind die potenziellen Offlinefähigkeit der Clients und die Möglichkeit,

diese Applikationen mit Werkzeugen wie beispielsweise PhoneGap in native Applikationen zu „verpacken“, die dann einerseits vollen Zugriff auf die Hardwarefunktionalitäten der Ausführungsumgebung haben, als auch wie native Apps durch die entsprechenden Distributionskanäle der Plattformen („App Stores“) verteilt werden können. Auch die Desktopwebanwendungen werden von diesem neuen Architekturmuster profitieren.

1.5 Entwicklungsprozess

Um eine effiziente und qualitativ hochwertige Entwicklung sicherzustellen, verwenden wir bei der Entwicklung erprobte Open-Source-Software-Komponenten wie beispielsweise jQuery Mobile als Komponentenbibliothek für die Gestaltung der Benutzeroberfläche oder AngularJS als JavaScript-Framework für die Entwicklung der JavaScript-basierten Clientschicht.

Durch den Einsatz testgetriebener Entwicklung stellen wir die Qualität und kontinuierliche Auslieferungsfähigkeit der Ergebnisse unserer Entwicklungsiterationen sicher. Wir nutzen dieses Vorgehen sowohl bei einzelnen Komponenten durch Unit Tests als auch bei der Entwicklung von Applikationsfunktionen durch automatisierte Akzeptanztests. Auch hier verwenden wir erprobte und weit verbreitete Werkzeuge wie beispielsweise das Behaviour-Driven-Design-Framework Jasmine und erweitern diese bei Bedarf um benötigte Komponenten (siehe das GitHub Projekt „Jasmine UI“).

Die Integration dieser Entwicklungspraktiken in einen Continuous-Integration-Prozess mit kontinuierlichen Projekt-Builds erlaubt eine feingranulare Überwachung des Zustands der Entwicklung und kurzfristige Reaktion auf Fehler und Probleme. Darüber hinaus ist damit die Einbettung der Entwicklung in eine agile, iterative Projektvorgehensweise möglich, welche die Bereitstellung auslieferbarer Ergebnisse in kurzen Zeitabständen erfordert.

1.6 Ziel des Buchs

Wir möchten hier keine Diskussion pro oder kontra native Entwicklung vs. Entwicklung von mobilen Web-Apps führen. Wie oben beschrieben, gibt es gute Argumente für beide Vorgehensweisen. Wir gehen in diesem Buch von der Annahme aus, dass Sie sich bewusst für den Weg der mobilen Web-Apps entschieden haben, und lernen möchten, wie man gutes Softwaredesign für eine JavaScript-Web-App implementiert. Ja, richtig. Vieles ist nicht rein Mobile-spezifisch, sondern gilt auch für nicht-mobile JavaScript-Web-Apps. Sie lernen also vielleicht mehr, als Sie eigentlich gedacht hatten...

Unser Ziel ist es nicht, aufs letzte Bit zu zeigen, was in HTML5 alles geht, also z. B. Media Queries, UI-Design, neue Tags, etc., dafür gibt es mittlerweile reichlich gute Literatur am Markt. Unser Anspruch ist es vielmehr, Ihnen testgetriebene Entwicklung für JavaScript näherzubringen, Sie in die Lage zu versetzen, wartbaren Code, Clean Code, zu schreiben und die Gesamtzusammenhänge zu verdeutlichen.

Wir werden keine Featureschlacht vorstellen, sondern einen kompletten End2End-Durchlauf über alle relevanten Implementierungsbestandteile einer mobilen Webapplikation mit Backend-Anbindung an einem einzigen, durchgängigen Beispiel beschreiben.

Kurz: Wir möchten Ihnen die Anleitung zum Einstieg in die Programmierung mobiler Web-Apps zur Verfügung stellen, die uns damals gefehlt hat, und Sie schnell in die Lage versetzen, architektonisch saubere, qualitativ hochwertige und dauerhaft stabile Web-Apps zu programmieren.

1.7 Zielgruppe dieses Buchs

Wir adressieren mit diesem Buch all diejenigen, die mobile JavaScript-Web-Apps professionell entwickeln möchten und sich dabei auch für innere Werte wie Wartbarkeit, Erweiterbarkeit und Testbarkeit interessieren. Wir sprechen im Wesentlichen die Zielgruppe der klassischen Java-Webentwickler an, da diese vermutlich alle aufgrund der schnell wachsenden Verbreitungszahlen von Smartphones und Tablets, kurz- bis mittelfristig Grundkenntnisse erlernen und erste Schritte mit dem neuen Paradigma der Web-Apps durchlaufen müssen. Aber auch für die Entwickler klassischer, nicht notwendigerweise mit Java erstellter Webanwendungen ist das Buch gedacht, da Java-Kenntnisse auf Frontend-Seite natürlich nicht nötig sind und hier somit zwei Welten zusammenwachsen. Wenn Sie also generell vor der Herausforderung stehen, reichhaltige JavaScript-Web-Apps im Enterprise-Umfeld schreiben zu wollen, dann lohnt es sich, nun weiterzulesen.

1.8 Vorstellung des durchgehenden Use Case

Das in diesem Buch verwendete Praxisbeispiel basiert auf dem bekannten Rent Your Legacy Car (RYLC) Showcase. Dieser wurde im Java Magazin in einer 14-teiligen Reihe (11.2008 bis 12.2009) verwendet, um SOA-Prinzipien zu verdeutlichen und anschließend im Sonderheft SOA Spezial zusammengefasst. In diesem Buch erweitern wir diesen Showcase um einen mobilen Vertriebskanal mit der Möglichkeit einer mobilen Autoanmietung.

1.8.1 Das Beispielunternehmen RYLC

Rent Your Legacy Car (RYLC) ist ein Autoverleih-Unternehmen mit der Vision, in den nächsten Jahren zu den drei umsatzstärksten Unternehmen des Car Rental Segments zu gehören. Seit Längerem ist das RYLC-Management mehr und mehr gezwungen, sich mit einer aggressiveren Marktsituation auseinandersetzen zu müssen. In den letzten Jahren hat RYLC immer wieder Marktanteile verloren. Die Konkurrenz konnte neue Produkte und besseren Service schneller bieten, während RYLC aufgrund von IT-Integrations-problemen nicht nachkam. Zur Adressierung dieser Hauptschmerzpunkte erarbeitet die Fachseite die folgenden Anforderungen:

Als Unique Selling Point (USP) soll eine neue, hochqualitative Dienstleistung eingeführt werden, die RYLC deutlich vom Mitbewerb differenziert. Firmen- als auch Privatkunden bekommen einen Mietwagen innerhalb einer Stunde garantiert zum gewünschten Einsatzort geliefert, der ohne Mehrkosten am Reiseziel wieder abgeholt wird. Falls der Kunde das Auto an einer Basisstelle selbst abholt oder zurückbringt, erhält er einen Rabatt.

Bisher war das Unternehmen ausschließlich über den Vertriebskanal DirektSales an zentralen Knotenpunkten wie Flughafen, Hauptbahnhof und Schiffshafen vertreten. In Zukunft werden weitere Vertriebskanäle wie das Smartphone, das Internet und SMS eingeführt.

Die garantierte Qualität von Leihwagen soll durch bessere Kontrolle des Fahrzeugzustands (Sauberkeit, Wartungsintervall) zur Verbesserung der Kundenzufriedenheit gesteigert werden.

Der Kontext in diesem Buch: Erweiterung der Vertriebskanäle um eine Möglichkeit zur Autoanmietung von unterwegs.

1.8.2 Fachlichkeit der Beispielanwendung

In diesem Buch werden wir sukzessive insgesamt vier RYLC Use Cases entwickeln:

- Login/Logout
- Benutzerprofil des angemeldeten Benutzers anzeigen
- Leihwagen-Buchungshistorie des angemeldeten Benutzers anzeigen
- Durchführen einer Leihwagen-Onlinebuchung von unterwegs

Am Ende des Buchs werden Sie die vollständig funktionsfähige RYLC-Web-App auf Ihrem eigenen Smartphone oder Tablet ausprobieren können. Auf diesem Weg werden wir alle relevanten Schritte erläutern, wie ein solches Ergebnis professionell erzeugt werden kann. Wie bereits beschrieben, legen wir den Fokus nicht auf eine vollständige Beschreibung der einzelnen Themen, sondern auf eine möglichst gute Erläuterung des Zusammenspiels aller für diesen Showcase nötigen Einzelaspekte.

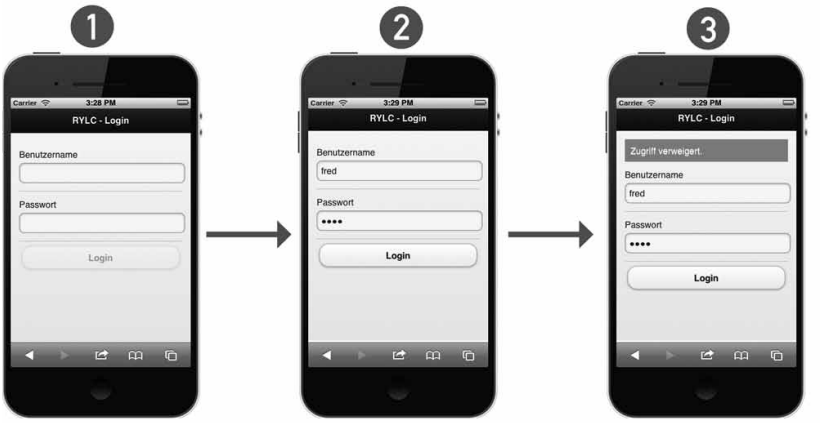

Abbildung 1.1: RYLC-Login mit falschem Passwort

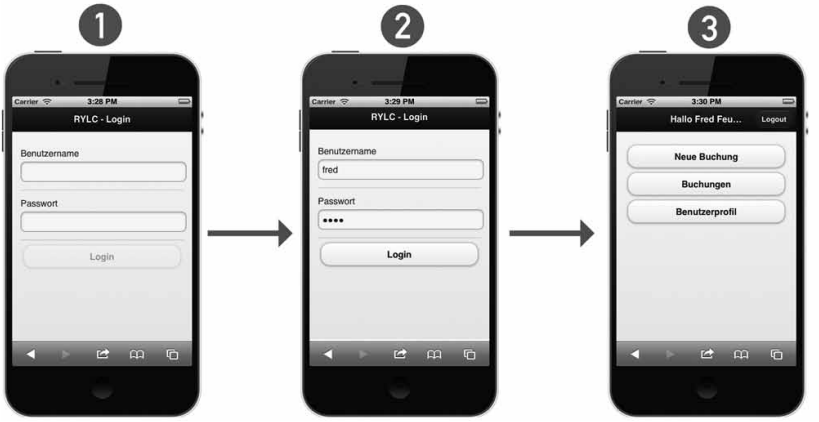

Abbildung 1.2: RYLC-Login mit korrektem Passwort

Abbildung 1.3: RYLC-Logout

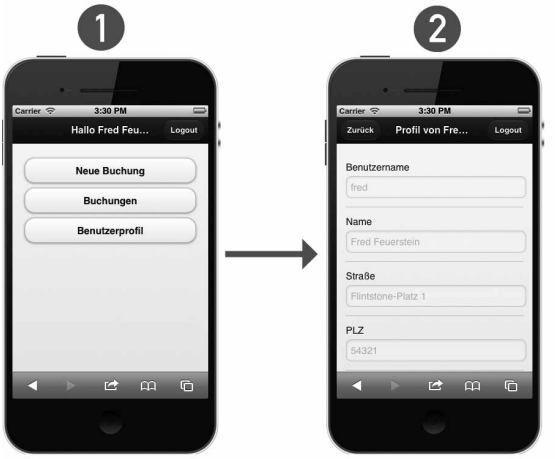

Abbildung 1.4: RYLC-Benutzerprofil

Abbildung 1.5: RYLC-Buchungshistorie