

Alle Neuerungen
von jQuery 3.0 S. 34

Nativer Code trifft
React Native S. 68

Graphdatenbank Neo4j:
Neue Version S. 84

web & mobile
DEVELOPER

web & mobile

webundmobile.de

DEVELOPER

Social Media integrieren

So nutzen Sie Social Media für eigene Applikationen ab S. 18



Funktionale Programmierung

Philip Ackermann erläutert die Prinzipien und Techniken der funktionalen Programmierung in JavaScript S. 40

Ausgabe 9/16

Deutschland
14,95 EUR

CH: 29,90 CHF
A, B, NL, L:
16,45 EUR



4 198255 014951 09

DB Systel stellt die Weichen für den IT-Arbeitsplatz der Zukunft



**Lesen Sie
die aktuelle
Case Study
über den Einsatz
von Citrix-
Technologien**

**GRATIS
Whitepaper**

Lesen Sie im kostenlosen Whitepaper von **CITRIX®**

- Effizienteres Management von IT-Arbeitsplätzen
- Einfachere Migration auf neue Betriebssysteme
- Endgeräteunabhängiger Zugriff auf Applikationen
- Schnellere Reaktionen auf neue Business-Anforderungen
- Neue Möglichkeiten für mobile Zugangsszenarien

Jetzt kostenlos downloaden:

<http://digital.internetworld.de/citrix-wp-dbsystel-wump/>

Ein Service von:

 **INTERNET WORLD Business**

Ihr Whitepaper bei uns?

Anfragen an sales@nmg.de

oder Tel. 089 / 7 41 17 – 124.



Soziale Netze

An den sozialen Netzen kommt heute kein Entwickler mehr vorbei. Durch die Integration dieser Netze in eigene Applikationen ergeben sich eine Menge Vorteile.

Soziale Netze sind im Web nicht nur allgegenwärtig, sie bieten Entwicklern inzwischen auch eine Fülle an überaus reizvollen Features. Dank einer durchdachten tiefen Integration ihrer Webanwendungen und mobilen Apps mit sozialen Netzen können sich Entwickler die Innovationskraft von deren Anbietern zu Eigen machen. Welche Möglichkeiten sich für Entwickler auf diesem Gebiet ergeben, zeigt ein umfangreicher Grundlagenartikel von Filipe Pereira Martins und Anna Kobylinska ab Seite 18.

»Soziale Netze bieten Entwicklern inzwischen eine Fülle an überaus reizvollen Features.«

Endlich ist es so weit: jQuery 3.0 ist da. Ein Versionssprung wie dieser ist eine gute Gelegenheit, alten Ballast über Bord zu werfen und auch Änderungen vorzunehmen, die nicht abwärtskompatibel sind. Die entscheidende Neuerung an jQuery 3.0 ist sicher, dass Deferred jetzt Promises/A+-kompatibel ist, aber es gibt auch weitere interessante Kleinigkeiten. Dr. Florence Maurice stellt die neue Version des beliebten JavaScript-Frameworks ab Seite 34 im Detail vor.

Mit Version 3 verfolgte Neo Technology das Ziel, vorhandene Systemgrenzen – vor allem die Größe der Graphen betreffend – zu beseitigen. So entschloss sich der Hersteller, für das Herzstück der Systemarchitektur, den Datenspeicher (Data Store), ein vollständiges Reengineering durchzuführen. Auf der Basis dieses Redesigns unterstützt Neo4j in der neuen Version erstmals Giant Graphs, also sehr große Graphen mit massenhaft vielen Knoten und Kanten. Alle Neuerungen von Version 3 der Graphdatenbank Neo4j beschreibt Frank Simon in einem Artikel ab Seite 84.

Die Basiskomponenten sind beim Single-Board-Computer Raspberry Pi alle auf einem Board, mit Zubehör kann man aber noch einiges mehr herausholen. Interessante Erweiterungen stellt Dr. Markus Stäuble in einem Artikel ab Seite 130 vor.

Ihr Max Bold
chefredakteur@maxbold.de



Philip Ackermann

erläutert, wie mit dem History API auf den Browserverlauf per JavaScript zugegriffen werden kann (S. 52)



Christian Bleske

stellt das iOS MapKit vor und zeigt, wie man damit Karten komfortabel in eigene Apps integrieren kann (S. 64)



Katharina Sckommodau

zeigt, wie mit Cloud Computing, SaaS und Online-Editoren Bildbearbeitung über das Netz funktioniert (S. 114)



INHALT

Update

Cloud-Monitor 2016

Public Cloud hängt Private Cloud ab

Feature

Deep Integration mit sozialen Netzen

Soziale Netze bieten Entwicklern inzwischen eine Fülle an überaus reizvollen Features

HTML, CSS & JavaScript

jQuery 3

Beim beliebtesten JavaScript-Framework aller Zeiten steht der nächste große Versionssprung an: jQuery 3

Funktionale Programmierung in JavaScript

Prinzipien und Techniken der funktionalen Programmierung in JavaScript

Das History API

Das History API ermöglicht den Zugriff auf den Browserverlauf per JavaScript

Mobile Development

Navigation mit Qt Quick Controls 2

Die neuen Qt Quick Controls 2 bieten viele Varianten der Navigation durch eine App an

iOS MapKit

Den Weg zu kennen kann nützlich sein. Mit iOS MapKit lässt er sich sogar in eine Karte zeichnen

React-Native-Bibliotheken (Teil 2)

Die innovative Architektur von JavaScript-Frameworks der zweiten Generation erlaubt Entwicklern die Erstellung von Apps, die mit geringem Aufwand auf native APIs zugreifen

Programmieren auf dem iPad

Mit der neuen App Swift Playgrounds kann man direkt auf dem iPad programmieren

Backend

Version 3 der Graphdatenbank Neo4j

Die aktuelle Version 3 der Graphdatenbank Neo4j bietet Neuerungen in vielen wichtigen Funktionsbereichen

6

18

34

40

52

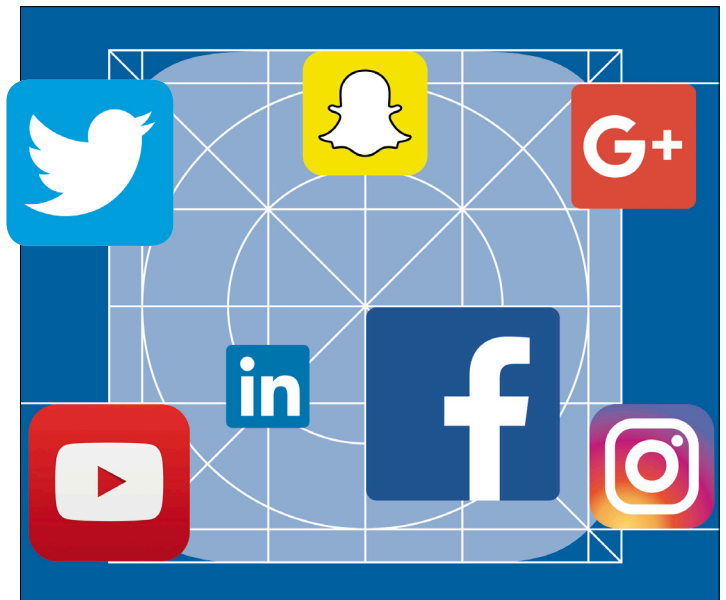
56

64

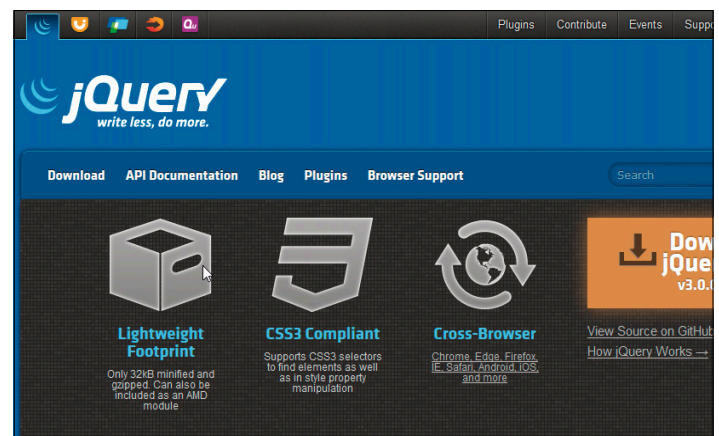
68

78

84



Soziale Netze sind im Web nicht nur allgegenwärtig, sie bieten Entwicklern inzwischen auch eine Fülle an überaus reizvollen Features für die Nutzung in eigenen Applikationen **S. 18**



jQuery zählt bei Entwicklern zu den beliebtesten JavaScript-Frameworks. Jetzt wurde die Version 3.0 vorgestellt **S. 34**

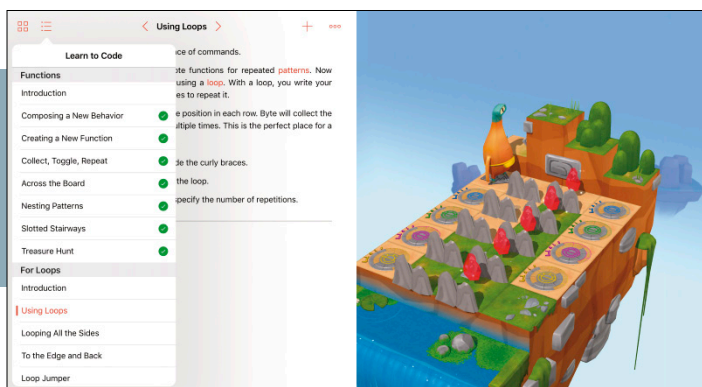
Experten in dieser Ausgabe



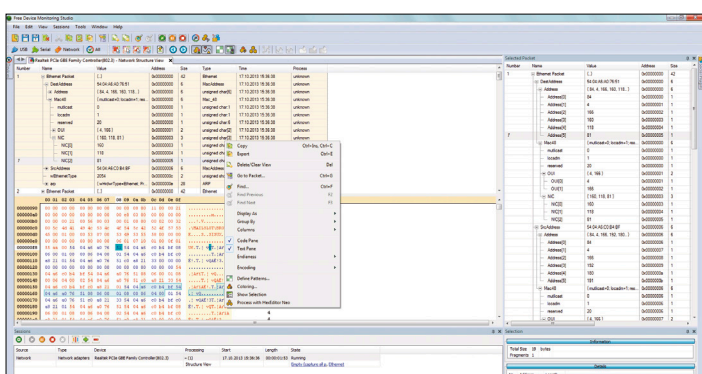
Philip Ackermann erläutert die Grundsätze der funktionalen Programmierung in JavaScript **40**



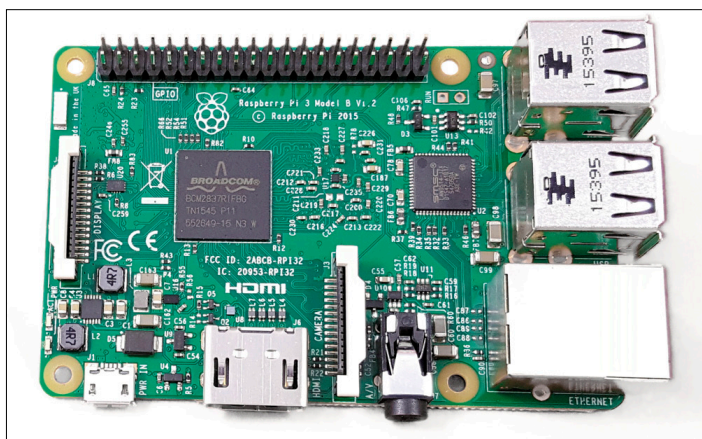
Ekkehard Gentz entwickelt eine ansprechende Navigation für Apps mit Hilfe von Qt Quick Controls 2 **56**



Zusammen mit iOS 10 – der neuen Version des Betriebssystems für iPhone, iPad und iPod touch – veröffentlicht Apple auch eine neue iPad-App namens Swift Playgrounds **S. 78**



Neo4j Version 3: Seit einigen Jahren verbreiten sich Graphdatenbanken zunehmend über verschiedene Anwendungsgebiete in immer mehr Unternehmen **S. 84**



Der Raspberry Pi 3 Modell B lässt sich mit allerhand sinnvollem Zubehör mächtig aufrüsten **S. 130**

Jetzt abonnieren

Sichern Sie sich jetzt die **web & mobile developer** im Jahresabo und profitieren Sie von exklusiven Services und Angeboten für Abonnenten.

<https://shop.webundmobile.de>

Web Application Security Scanning mit Nikto

Webbasierte Applikationen bilden vielfach das Herzstück der IT-Infrastruktur eines Unternehmens und müssen daher optimal gesichert werden **98**

Software testen in der Cloud

Cloud-Plattformen können Kosten senken und die Entwicklung beschleunigen **102**

Twig: Template-Engine für PHP

Mit der Bibliothek Twig lässt sich Trennung von Skriptcode und HTML unter PHP leicht realisieren **108**

Offene Standards vereinfachen die Integration externer Daten

Die Einbindung externer Datenquellen gelingt besser, wenn ein offener Standard für Datenkonnektivität wie OData eingesetzt wird **112**

Beyond Dev

Grafik für Entwickler: Cloud

Cloud Computing, Software as a Service und Online-Editoren – wie Bildbearbeitung über das Netz funktioniert **114**

Developer Week 2016

Informationen satt bei bester Stimmung und vielen interessanten Gesprächen **118**

State of the Application

Gleich im doppelten Sinne geht es um Status. Wie ist der Stand der Dinge in Sachen Zustandsverwaltung? **122**

Nützliches Zubehör für den Raspberry Pi

Die Basiskomponenten sind beim Single-Board-Computer Raspberry Pi alle auf einem Board – mit Zubehör kann man aber noch einiges mehr herausholen **130**

Standards

Editorial **3**

Impressum **139**

Online-Recht **140**

Arbeitsmarkt **142**

Dienstleisterverzeichnis **145**

Vorschau **146**

NEWS & TRENDS

AKTUELLE NEWS AUS DER ENTWICKLERSZENE

CloudBees

Neuer Report zu Docker, Jenkins und CD

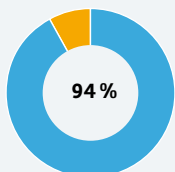
Der Anbieter von Continuous-Delivery-Lösungen CloudBees hat die Ergebnisse einer aktuellen Umfrage zum Thema »The State of Docker, Jenkins and Continuous Delivery« bekannt gegeben.

Die Umfrage wurde anlässlich der DockerCon in Barcelona durchgeführt und besagt, dass Unternehmen sich ihres Bedarfs an einer offenen Plattformlösung bewusst sind, mit der sie die Einführung von Continuous-Delivery-Prozessen (CD-Prozessen) vereinfachen und beschleunigen können. Mit Stand vom Juni 2016 erfolgten mehr als 5 Millionen Abrufe des Jenkins-Images aus der Docker-Registry. In der Umfrage gaben 83 Prozent der Befragten an, dass sie Docker-Container für die Ausführung von Jenkins-Build-Servern einsetzen wollen – ein Indiz dafür, dass Jenkins und Docker zu den Standardtechnologien werden, die für eine zeitgemäße Realisierung von Software-Anwendungen erforderlich sind.

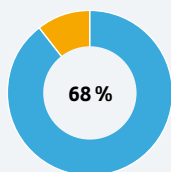
Die Umfrage wurde unter den Besuchern der DockerCon Barcelona verteilt und von diesen zwischen November und Dezember 2015 beantwortet. Zwei Drittel der antwortenden Teilnehmer an der Umfrage gaben ihre Funktion mit Software-Entwickler, DevOps-Anwender oder Software Architect an, während das übrige Drittel andere Funktionen wie Gruppenleiter, Führungskraft, Build Manager und QS nannten.

https://pages.cloudbees.com/Website_Docker-Jenkins.html

Docker Container Adoption

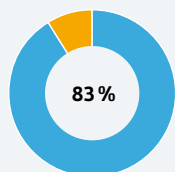


... of survey respondents will run new application in Docker containers in both test and production.

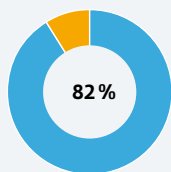


... of survey respondents have already adopted Docker or plan to adopt Docker in the next 12 months.

Shipping CI/CD pipelines with docker containers



... of survey respondents want to use Docker containers for running Jenkins build servers.



... of survey respondents want to deliver application infrastructure using Docker.

web & mobile developer 9/2016

Quelle: CloudBees

Cloud-Monitor 2016

Public Cloud hängt Private Cloud ab

54 Prozent der Unternehmen in Deutschland haben 2015 Cloud Computing eingesetzt. Das besagt eine repräsentative Umfrage von Bitkom Research im Auftrag von KPMG unter 457 Unternehmen ab 20 Mitarbeitern. Im Vorjahr hatte die Cloud-Quote noch bei 44 Prozent gelegen. Bemerkenswert ist: Der Zuwachs speist sich vor allem aus kleinen und mittelgroßen Unternehmen. Die Cloud-Nutzung in Unternehmen mit 100 bis 1999 Mitarbeitern stieg um 7 Prozentpunkte auf 62 Prozent und in Unternehmen mit 20 bis 99 Mitarbeitern sogar um 11 Prozentpunkte auf 52 Prozent. Bei Unternehmen ab 2000 Mitarbeitern legte die Nutzung auf hohem Niveau nur um einen Prozentpunkt auf 69 Prozent zu. Axel Pols, Geschäftsführer von Bitkom Research: »Der Mittelstand hat seine Zurückhaltung beim Cloud Computing endgültig abgelegt.«

Der Umfrage zufolge nutzten 2015 rund 26 Prozent der Unternehmen Public-Cloud-Computing, im Jahr zuvor waren das erst 16 Prozent. Dagegen stabilisierte sich der Einsatz von Private Clouds bei 38 Prozent (Vorjahr: 39 Prozent). »Bislang installierten die Unternehmen vor allem Private Clouds, weil vielen die Nutzung über das öffentliche Internet zu unsicher schien. Das ändert sich jetzt«, erläutert Pols diese Entwicklung. »Das vergangene Jahr markiert den Durchbruch für

Public Cloud Computing in der deutschen Wirtschaft.«

Vor allem die großen US-Cloud-Anbieter hätten nach der Geheimdienstaffäre auf die Sicherheitsbedenken der Anwender reagiert. »Viele Cloud-Provider haben massiv in die Sicherheit investiert und Rechen-



Foto: Bitkom Research

Axel Pols, Geschäftsführer Bitkom Research

zentren in Europa und speziell in Deutschland aufgebaut«, sagt Peter Heidkamp, Partner und Head of Technology bei KPMG.

www.bitkom-research.de/CM2016

Software Heritage

Ein Museum für Quellcode

Das Projekt Software Heritage will den kompletten Quellcode der Menschheit zusammentragen. Im Gegensatz zu in Stein gravierten Schriftzeichen sei Software allzu vergänglich und korruptierbar.

Wer Sourcecode im Internet veröffentlicht hat oder dies tun wird, ist sicher dabei. Zwar wurde das Projekt Software Herita-

Zahl des Monats

Die IT marschiert in die Cloud: **92 %** der neu gegründeten Internetfirmen gehen laut Umfrageergebnissen von Anfang an in die Cloud. Beinahe die Hälfte der mittelständischen Wirtschaft (**48 %**) ist demnach ebenfalls bereits auf dem Weg in die Cloud.

Quelle: www.leaseweb.de

ge gerade erst öffentlich angekündigt, es ist aber bereits seit rund einem Jahr in Arbeit. Über



Software Heritage plant eine moderne Bibliothek von Alexandria für Software

22 Millionen Softwareprojekte hat das Team bereits zusammengetragen und dabei mehr als 2,6 Milliarden Dateien mit Quellcode archiviert. Hinter dem Projekt Software Heritage steht das nationale französische Institut für Informatik und Automatisierung. Schon die jetzigen Datenmengen zeigen, dass es sich bei der gewählten Aufgabe um ein Mammutprojekt handelt. Deshalb will sich das Projekt eine verteilte Infrastruktur geben und sucht dafür Mitstreiter, die einerseits weiteren Code zur Sammlung beitragen und andererseits beim Sicherstellen der Verfügbarkeit des Archivs mithelfen.

www.softwareheritage.org

Optimierungs-Index

Firmen verschenken SEO-Potenzial

»Welche SEO-Potenziale schlummern in Deutschlands Unternehmenswebsites«, wollte die Mindshape GmbH wissen und hat deshalb mehr als 3500 Internetauftritte aus 15 Branchen untersucht. Um die SEO-Gestaltung vergleichbar zu ma-

chen, hat Mindshape einen Index entwickelt, der den Optimierungsgrad einer Website ausdrückt. Der durchschnittliche Index für alle untersuchten Unternehmen stieg im Vergleich zum Vorjahr lediglich von 0,37 auf 0,39. Demnach besteht nach wie vor ein großes Verbesserungspotenzial bei Online-Marketing und Suchmaschinenoptimierung.

Fortschritte konnten vor allem in puncto Technik beobachtet werden. Auffällig ist, dass deutlich mehr Unternehmen über eine responsive Website verfügen als im Vorjahr. 2016 waren das 27 Prozent, 2015 nur 15 Prozent. Andererseits ignorieren immer noch 42 Prozent komplett die Datei robots.txt als Instrument der Crawler-Steuerung.



Foto: Mindshape

Sebastian Erlhofer:

Geschäftsführer Mindshape

»Auch 2016 haben deutschsprachige Unternehmen die Bedeutung der Suchmaschinenoptimierung im Marketing-Mix noch nicht verinnerlicht«, resümiert Mindshape-Geschäftsführer Sebastian Erlhofer.

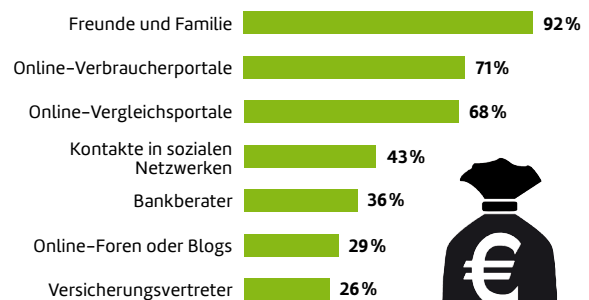
www.mindshape.de

Verbraucher vertrauen auf Online-Portale

Ob Smartphone-Tarif oder Steuererklärung, Kfz-Versicherung oder Baufinanzierung

Wenn es um das eigene Geld geht, ist guter Rat teuer. Doch wem vertrauen die Verbraucher in Finanzangelegenheiten am meisten? Diese Frage wurde in einer repräsentativen Studie im Auftrag des Digitalverbands Bitkom untersucht. Dabei sollten die Befragten angeben, für wie vertrauenswürdig sie verschiedene Personengruppen, Online-Dienste oder Institutionen halten, wenn es um finanzielle Ratschläge geht.

Inwiefern halten Sie die folgenden Institutionen oder Personengruppen für vertrauenswürdig, wenn es um finanzielle Ratschläge geht?



Wenig Vertrauen in den Rat von Versicherungsvertretern

web & mobile developer 9/2016

Quelle: Bitkom Research

Voraussetzung dafür war, dass sich die Befragten eine Beurteilung selbst zutrauen, weil sie zum Beispiel die Angebote kennen oder zumindest davon gehört haben. Gleich nach Freunden und Familie, die von 92 Prozent als vertrauenswürdig eingestuft werden, folgen Internetangebote. 71 Prozent vertrauen Online-Verbraucherportalen wie Finanzen.net oder Finanztip.de in Finanzfragen, und 68 Prozent halten Online-Vergleichsportale wie zum Beispiel Verivox.de oder Check24.de in dieser Hinsicht für vertrauenswürdig. 43 Prozent vertrauen zudem ihren Kontakten in sozialen Netzwerken bei Finanzangelegenheiten, gut ein Drittel setzt auf Bankberater (36 Prozent). Es folgen Online-Foren oder Blogs (29 Prozent) sowie Versicherungsvertreter (26 Prozent).

Bitkom rät, bei allen Informationen rund um Finanzfragen zu prüfen, welche Personen beziehungsweise Organisation diese verantworten. Nicht selten finden sich hier einzelne Unternehmen mit speziellen wirtschaftlichen Interessen.

www.bitkom.org

Progress-Studie

Zaghafte digitale Transformation

Eine aktuelle Studie von Progress zeigt, dass viele Unternehmen die digitale Transformation nicht mit der gebotenen Entschlossenheit angehen. In Deutschland ist diese Haltung zwar deutlich weniger verbreitet, kommt aber immer noch signifikant häufig vor. Praktisch alle Unternehmen schreiben der digitalen Transformation eine große Bedeutung zu. Bei der eigenen Digitalisierung stoßen sie aber häufig auf Hindernisse wie mangelnde Abstimmung oder kulturelle Widerstän-



Die digitale Transformation wird zwar als wichtig erkannt, stößt bei der Umsetzung jedoch häufig auf Hindernisse

de. Viele von ihnen haben das Gefühl, dass die Zeit drängt. Zu diesen Ergebnissen kommt die Studie »Are Businesses Really Digitally Transforming or Living in Digital Denial?«.

»Digitale Technologien verändern das Geschäft von Unternehmen radikal, und die treibende Kraft dahinter ist die Customer Experience«, sagt Mark Troester, Vice President Digital Solutions bei Progress.

www.progress.com

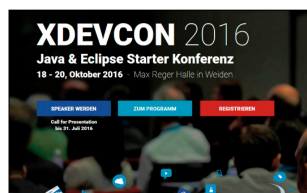
XDEVCON**Call for Papers für die Java Starter Konferenz**

Bis Ende August haben Java-, Web- und App-Entwickler Zeit, sich mit Vorschlägen für einen Vortrag auf der XDEVCON 2016 zu bewerben.

Auf der XDEVCON dreht sich alles um den schnellen Java- und Eclipse-Einstieg. In zahlreichen Live-Demos erfahren die Teilnehmer, wie man HTML5-Webanwendungen, Mobile Apps und Cloud-fähige Anwendungen mit Java und Eclipse entwickelt und welche neuen Tools die Anwendungsentwicklung mit Eclipse stark vereinfachen und beschleunigen.

Das Besondere an der XDEVCON ist, dass die Teilnehmer keine Java-Vorkenntnisse besitzen müssen. Wichtige Java-Grundlagen werden leicht verständlich vermittelt und die Teilnehmer erfahren, welche Themen für Umsteiger besonders wichtig sind.

Die Konferenz richtet sich vor allem an IT-Abteilungen, die ein Java-Team aufbauen und unternehmenskritische Altanwendungen möglichst schnell und kostengünstig auf Java migrieren möchten. Die XDEVCON

**Entwicklerkonferenz:**

Im Oktober findet die XDEVCON für Java-, Web- und App-Entwickler statt

wird zum 6. Mal ausgetragen und findet heuer vom 18. bis 20. Oktober 2016 in Weiden statt.

Mit Unterstützung der **web & mobile developer** kostet die Teilnahme am Starter Day nur 19 Euro. Der Eintritt für die

nachfolgende Developer Night ist sogar kostenlos. Mit unserem Heft-Code WMD-XDC2016 können Sie zudem ganze 100 Euro bei der Anmeldung für die Gesamtkonferenz sparen.

www.xdevcon.de

EGroupware**Neuprogrammierung abgeschlossen**

Das Open-Source-Projekt EGroupware hat seine Anwendung vollständig neu programmiert, um zeitgemäßen Anforderungen an die Nutzung von Groupware zu entsprechen.

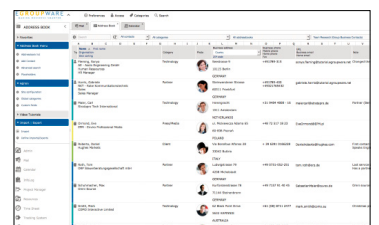
JavaScript wird es künftig erleichtern, die browserorientierte EGroupware um zusätzliche Module und Gestaltungselemente zu erweitern.

Auf den kleineren Bildschirmen von Mobilgeräten erscheint EGroupware nicht mehr wie früher als eine Verkleinerung der Desktop-Oberfläche. Vielmehr gibt es eine reduzierte Darstellung, die allerdings jederzeit alle Einträge anzeigen kann. Farben führen zu den verschiedenen Anwendungen. Eine App-Übersicht erleichtert den Wechsel zwischen den Anwendungen. Sie werden mit den bei Smartphones und Tablets üblichen Gesten oder über Buttons bearbeitet. Dabei synchronisiert das sogenannte Mobil-Templat im Hintergrund nicht jeweils die einzelnen Anwendungen, sondern alle Zugriffe erfolgen in Echtzeit.

Als letztes der Kernmodule wurde der Kalender von EGroupware komplett neu entwickelt, um seine Nutzung übersichtlicher und einfacher zu machen. Ein Klick in der Toolbar wechselt zwischen Tages-, Wochen- und Monatsansicht. Weniger häufig benötigte Ansichten können Anwenden aus der Toolbar heraus ziehen und ablegen.

Den Kategorien der Einträge können Teams unterschiedliche Farben zuordnen, was die Terminkoordination übersichtlicher macht. Wenn sich durch organisatorische Neuerungen Zuständigkeiten verändern, lassen sich die Termine per Drag and Drop zwischen verschiedenen Kalendern verschieben.

Über alle Anwendungen hinweg ist die EGroupware-Bedienoberfläche auch in der Desktop-Version einheitlicher und wirkt leichter, weil sie übersichtlicher ist. Das seitlich angeordnete Menü lässt sich ausschalten, sodass auch auf kleineren Bildschirmen mehr Platz für die Einträge sichtbar bleibt.



Die Usability auf Desktops, Tablets und Smartphones stand bei der Neuprogrammierung im Vordergrund

EGroupware bleibt beim Wechsel zwischen den verschiedenen Apps bei der von den Benutzern zuletzt präferierten Darstellung mit oder ohne Seitenmenü. Außerdem sind die Funktionselemente in allen Anwendungen einheitlich angeordnet.

»Mit der Version 16.1 hat das Open-Source-Projekt EGroupware einen Meilenstein erreicht«, erklärt Ralf Becker, Initiator und Koordinator des Open-Source-Projekts. »Funktional und auch hinsichtlich der technischen Grundlagen bietet EGroupware exzellente Voraussetzungen zur Integration in bestehende Umgebungen und für künftige Erweiterungen. Die Usability ist auf Desktops, Tablets und Smartphones gleichermaßen vorbildlich.«

www.egroupware.org

Jetzt kostenlos testen!



2x
gratis!



Praxiswissen für Entwickler!

Testen Sie jetzt 2 kostenlose Ausgaben und erhalten Sie exklusiven Zugang zu unserem Archiv.

webundmobile.de/probelesen

DataCore

Speicher-Performance: Neuer Weltrekord

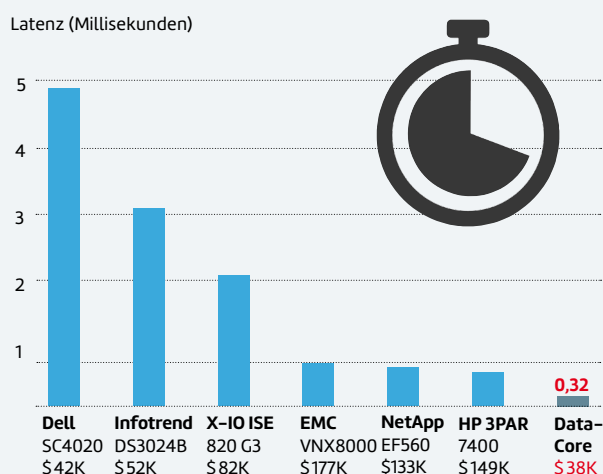
DataCore hat einen neuen Weltrekord für die Durchsatzleistung von externen Speichersystemen gemäß Storage Performance Council (SPC) aufgestellt. Der DataCore Parallel Server durchbrach dabei erstmals die 5-Millionen-IOPS-Schallmauer (5.120.098,98 SPC-1 IOPS) und setzt sich mit weitem Abstand an die Spitze der Top Ten der SPC-1-Performance-Liste. Der erzielte Wert bedeutet laut DataCore mehr I/O-Operationen pro Sekunde als bei den beiden bisher Führenden zusammen.

DataCores Rekordergebnis wurde in einer hyperkonvergenten Konfiguration auf zwei Standard-Lenovo-Servern erzielt, die über Fibre Channel mit zwölf externen Hosts verbunden waren, die den notwendigen I/O-Verkehr über anspruchsvolle Unternehmens-Applikationen generierten. Diese 2-Knoten-Konfiguration arbeitete mit dem DataCore Parallel Server, einer Softwarelösung, die das moderne Multicore-Computing nutzt, um I/O-Prozesse parallel zu verarbeiten, damit also Standard-Rechner in hochperformante Server powered by DataCore Parallel-I/O-Technologie verwandelt. Diese sind bestens gerüstet für anspruchsvolle Anwendungen mit besonderen Anforderungen an die I/O-Leistung sowohl im Speicher-Netzwerk als auch bei hyperkonvergenten Einzelsystemen.

Die durchschnittliche Antwortzeit unter Volllast ist ein weiterer aussagekräftiger Indikator für den Nutzen eines Systems, insbesondere für Anwender, bei denen es darum geht, schnell große Datenmengen auszuwerten und damit einen Wettbewerbsvorteil zu erzielen. Die DataCore-Parallel-Server-Software erreichte im Durchschnitt eine Antwortzeit von 280 Mikrosekunden (0,28 ms) während des achtstündigen SPC-1-Dauertests. Die bislang führenden Systeme von Huawei und Hitachi zeigten mit 0,92 und 0,96 Millisekunden im Schnitt dreimal langsamere Antwortzeiten.

www.datacore.de

DataCore hat die niedrigste Latenzzeit.



Der DataCore Parallel Server stellte einen neuen Performance-Weltrekord auf

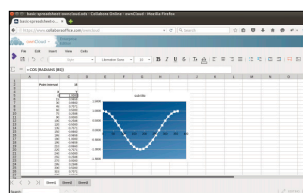
web & mobile developer 9/2016

Quelle: DataCore

ownCloud und Collabora

LibreOffice Online in ownCloud Enterprise

Collabora Online ist ein wichtiger Baustein für ownCloud. Nutzer in Unternehmen erhalten damit endlich die häufig nachgefragte Open-Source-Bürosoftware, die alle wichtigen Dokumenten-, Tabellen- und Präsentationsformate unterstützt. Sie können damit die Produktivität erhöhen und



Collabora Online macht die Funktionen von LibreOffice Online für den produktiven Einsatz zugänglich

behalten gleichzeitig die volle Kontrolle über sensible Unternehmensdaten.

Teams können mit Hilfe von Shared Editing zusammenarbeiten. Wenn ein Anwender an einem Dokument arbeitet, können alle anderen die Änderungen in Echtzeit verfolgen. Aktuelle Nutzer von ownCloud können damit möglichen Versionskonflikten aus dem Weg gehen. Wenn mehrere Anwender unabhängig voneinander am selben Dokument arbeiten, erfolgt die Zusammenarbeit jetzt online innerhalb des Web-Frontends von ownCloud.

»Wir freuen uns sehr, die erste Lösung für LibreOffice Online überhaupt in Partnerschaft mit ownCloud zu veröffentlichen«, so Michael Meeks, General Manager bei Collabora Productivity. »ownCloud bringt Unternehmen innovative Möglichkeiten der Collabora-

tion. Es ist ein Privileg, gemeinsam mit ihren etablierten Support- und Wartungsteams die Anforderungen ihrer Kunden zu erfüllen.«

»Collabora hat unglaublich hart daran gearbeitet, die erste Open-Source-Suite zur Online-Bearbeitung in die Cloud zu bringen. Wir sind froh, die Lösung als Erste auf den Markt bringen zu dürfen«, so Holger Dyroff, Geschäftsführer von ownCloud. »Diese Integration zweier führender Projekte wird große Auswirkungen auf unsere Kunden haben. Wir blicken zuversichtlich in die Zukunft.« Die Lösung ist ab sofort unter dem Namen Collabora Online für ownCloud Enterprise bei ownCloud erhältlich.

www.owncloud.com/de

Software AG

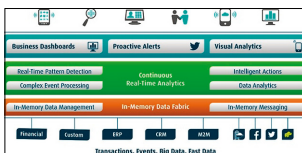
Community Edition für Apama

Die Software AG hat die Verfügbarkeit von Apama Community Edition angekündigt, einer kostenlosen Download-Version der Apama-Plattform für Streaming-Analytics, mit der Entwickler Streaming-Analytics-Anwendungen entwickeln und betreiben können. Apama Community Edition beinhaltet einen Großteil der Funktionalität und Eigenschaften der Premium-Version von Apama und ist für kleine Projekte und Anwendungen gedacht.

Die Software AG verbindet die Freigabe der Apama Community Edition mit dem Start einer neuen Online-Community für Kunden, Interessenten und Apama-Fans, die sich dort beteiligen und miteinander interagieren können. Das Apa-

ma-Team der Software AG wird in der Community Best Practices und Tipps aus seiner langjährigen Erfahrung mit der Entwicklung und Unterstützung von Apama sowie Lösungsvorschläge für viele unterschiedlich gelagerte Geschäftsprobleme bereitstellen, die den Nutzern der Community helfen, die Möglichkeiten von Apama ganz auszuschöpfen. Die neue Community ist unter www.apamacommunity.com zu erreichen.

Apama ist die marktführende Plattform für Streaming-Analytics und intelligente, automatisierte Entscheidungen auf der Basis schnell fließender Big Data und ermöglicht die Transformation relevanter Erkenntnisse in wirkungsvolle Geschäftsmaßnahmen. Das Produkt bietet Funktionen für Ereignisverarbeitung, Nachrichtenübermittlung



Mit der Community Edition können Entwickler Streaming-Analytics-Anwendungen erstellen

sowie In-Memory-Datenmanagement und Visualisierung.

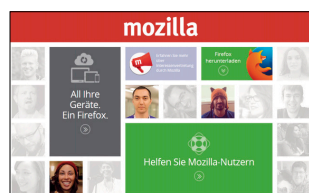
www.softwareag.com

Mozilla

Mozilla unterstützt Software-Entwickler

Mozilla unterstützt im Rahmen des Mission-Partners-Programms Open Source Secure Support (MOSS) acht Begünstigte aus den unterschiedlichsten Fach- und Technologie-Bereichen.

Über viele Jahre hinweg mussten Menschen mit Sehbeeinträchtigungen und vollständig Erblindete hohe Kosten in Kauf nehmen, wenn sie mit Windows-basierten Computern online gehen wollten. Die marktführende Software für sogenannte Bildschirmleser kostet weit über 1000 US-Dollar – dieser hohe Preis stellt zweifellos ein erhebliches Hindernis für ein offenes Web dar, das für alle zu-



Das Mission-Partners-Programm Open Source Secure Support (MOSS) von Mozilla unterstützt Entwickler aus den unterschiedlichsten Fach- und Technologie-Bereichen

gänglich sein soll. Aus diesem Grund hat das NVDA Project einen Bildschirmleser auf Open-Source-Basis entwickelt, der kostenlos zum Download verfügbar ist. Auch mit Firefox funktioniert der Reader sehr gut. Damit steht die Arbeit von NVDA in Übereinstimmung mit einem der im Mozilla-Manifest verankerten Prinzipien. Deshalb hat Mozilla beschlossen, das Projekt im Rahmen des neuen Mozilla Open Source Secure Support (MOSS) Mission Partners Awards mit 15.000 US-Dollar zu unterstützen. NVDA ist einer von acht Begünstigten aus den unterschiedlichsten Fach- und Technologie-Bereichen.

Für 2016 steht Mission Partners ein Budget von rund 1,25 Millionen US-Dollar zur Verfügung.

www.mozilla.org

eco-Umfrage

Deutsche haben Interesse an öffentlichem WLAN

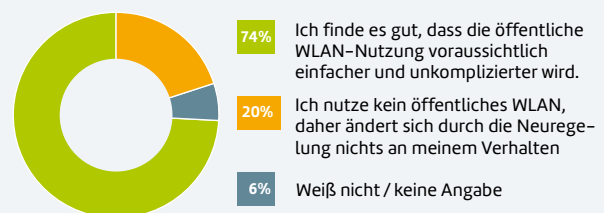
Die überwältigende Mehrheit der Deutschen freut sich über das Ende der Störerhaftung. Nach einer Umfrage des eco-Verbands finden es 74 Prozent der Befragten gut, dass die öffentliche WLAN-Nutzung voraussichtlich einfacher und unkomplizierter wird. Der Bundestag hat Anfang Juni das monatelang hart umkämpfte WLAN-Gesetz verabschiedet und damit das Ende der sogenannten Störerhaftung besiegelt. Aus Sicht von eco ist der Weg frei für öffentliche Hotspots in Deutschland. »Wir gehen davon aus, dass die Rechtslage nun geklärt ist und WLAN-Anbieter nicht mehr der Gefahr ausgesetzt sind, für rechtswidriges Verhalten ihrer Nutzer haften zu müssen«, sagt Oliver Süme, eco-Vorstand Politik & Recht.

Fast die Hälfte aller Befragten will öffentliches WLAN künftig stärker nutzen. 48 Prozent der Befragten sehen beim Ausbau des WLAN-Angebots im öffentlichen Raum den Staat in der Pflicht: Sie sind der Ansicht, dass Städte und Kommunen eigene WLAN-Netze anbieten sollten. 13 Prozent sehen den WLAN-Ausbau als eine alleinige Angelegenheit der Wirtschaft.

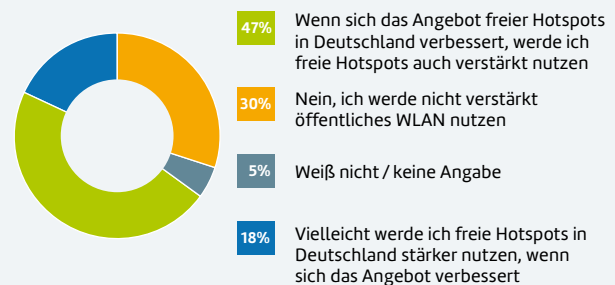
www.eco.de

eco-Umfrage: Das Haftungsrisiko fällt weg – Weg frei für offenes WLAN in Deutschland

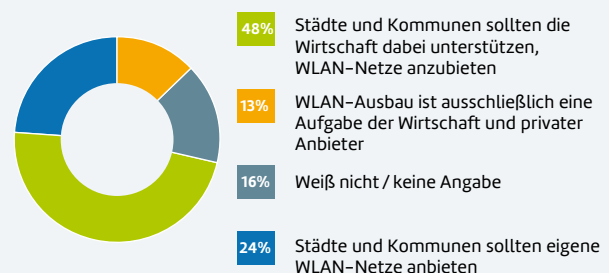
Neue Rechtslage: Deutsche freuen sich auf einfachere WLAN-Nutzung.



Freie Hotspots in Deutschland: Knapp die Hälfte der Nutzer will öffentliches WLAN stärker nutzen.



WLAN-Ausbau: Städte und Kommunen sollten sich stärker engagieren.



web & mobile developer 9/2016

Quelle: YouGov Deutschland AG

Prognose

Roboter-Boom im deutschen Mittelstand

81 Prozent der Automatisierungs-Manager in Deutschland erwarten, dass der Einsatz von Robotern in mittelständischen Betrieben stark zunehmen wird. Wichtiger Treiber ist eine neue Generation von Leichtbaurobotern, die kostengünstiger als klassische Industrieroboter sind und mit den Werkern ohne Schutzkäfing Schulter an Schulter zusammenarbeiten (COBOTS).

85 Prozent sind überzeugt, mit solchen Mensch-Roboter-Teams die Einsatzfelder der Automatisierung zu erweitern. Digital vernetzte Industrieroboter werden die Fertigungsmethoden revolutionieren – sind 86 Prozent überzeugt. Das sind Ergebnisse des Automatica Trend-Index 2016. Dazu wurden 100 Unternehmensentscheider befragt, die in ihrem Betrieb für den Einsatz von Automatisierungstechnik zuständig sind.

Die Weichen für den Kollegen Roboter werden in der Mehrzahl der deutschen Betriebe bereits gestellt: 52 Prozent haben entweder schon mit Industrierobotern automatisiert oder analysieren derzeit konkret die Einsatzmöglichkeiten von Robotertechnik für das eigene Unternehmen. 27 Prozent sind grundsätzlich an den neuen Lösungen interessiert. Fünf Prozent haben sich gegen den Einsatz von Industrierobotern entschieden. Stückkostenfragen und die einfache Bedienbarkeit stehen bei den Entscheidern hoch im Kurs, wenn es um die Frage nach dem Einsatz kollaborativer Roboter geht, die mit den Werkern Hand in Hand zusammenarbeiten. So hält es die große Mehrheit für wichtig, dass die COBOTS als Leichtbauroboter in der Anschaffung günstiger sind als klassische Industrieroboter (88 Prozent).

www.automatica-muenchen.de

Frage: Bitte geben Sie Ihre Prognose zum Einsatz moderner Industrieroboter ab.

Digital vernetzte Industrieroboter werden Fertigungsmethoden und -verfahren revolutionieren.



Industrieroboter werden die Schlüsseltechnologie für die industrielle Automatisierung sein (Industrie 4.0).



Mit den neuen kollaborativen Robotern erweitern sich die Einsatzfelder für die Automatisierung.



Der stark verbreitete Einsatz von Industrierobotern in mittelständischen Betrieben ist keine Frage des »ob«, sondern des »wann«.



■ stimme voll zu
 ■ stimme eher nicht zu
 ■ kann ich nicht beurteilen
■ stimme eher zu
■ stimme gar nicht zu

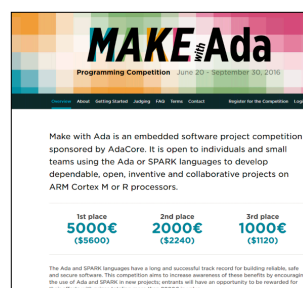
Industrieroboter sind nach Ansicht der meisten Automatisierungs-Manager in Deutschland auf dem Vormarsch

web & mobile developer 9/2016 n=100; Angaben in Prozent
Quelle: Messe München / Connection Global Competence

Embedded-Systeme

AdaCore startet Wettbewerb

Preise von insgesamt 8000 Euro sind beim Programmierwettbewerb »Make with Ada« von AdaCore zu gewinnen. Entwickler von Embedded-Systemen können dabei zeigen, wie sie zuverlässige und sichere Software mit Ada programmieren. Bis 30. September 2016 können Software-Entwickler mit Code, der unter Verwendung von Ada und



Beim Wettbewerb »Make with Ada« können Entwickler mit Ada-Kenntnissen Preise gewinnen

SPARK erstellt wurde, Preise gewinnen. Der Wettbewerb steht Einzelpersonen und Teams mit bis zu vier Mitgliedern offen. Ziel ist, ein Embedded-Software-Projekt für einen ARM-Cortex-M- oder -R-Prozessor zu entwerfen und zu implementieren, bei dem Ada und/oder SPARK als hauptsächliche Programmiersprachen verwendet werden. Die Teilnehmer müssen nachweisen, dass ihr System den Anforderungen entspricht und mit fundierten Software-Engineering-Methoden entwickelt wurde. Die Preisträger werden im November 2016 bekannt gegeben.

Die Jury umfasst namhafte Experten für Embedded-Systeme, so Jack Ganssle, Principal Consultant bei

The Ganssle Group, Dick Selwood, Europa Editor bei TechfocusMedia, Bill Wong, technischer Redakteur bei Penton Media, und Cyrille Comar, President von AdaCore.

<http://makewithada.org/getting-started>

Red Hat und Eurotech

IoT-Cloud-Plattform-Projekt

Red Hat und Eurotech, ein Anbieter von Machine-to-Machine-Plattformen (M2M) und Internet-of-Things-Lösungen (IoT), starten ein neues Open-Source-Eclipse-Foundation-Projekt. Des- sen Ziel ist die durchgehende Verwaltung von IoT-Edge-Devices, angefangen von der Connectivity und der Konfiguration bis zum gesamten Applikations-Lebenszyklus. Kombiniert mit dem bereits existierenden Projekt Eclipse Kura bietet



Das gemeinsam von Red Hat und Eurotech gesponserte Projekt Eclipse Kapua basiert auf der Eurotech Everyware Cloud

das gemeinsam gesponserte Projekt Eclipse Kapua IoT-Entwicklern und Anwendern eine offene Plattform zur End-to-End-Implementierung von IoT-Lösungen. Sie vermeiden damit eine kostspielige Herstellerabhängigkeit und profitieren von der Community-getriebenen Weiterentwicklung.

www.redhat.com

Jetzt kostenlos testen!



**2x
gratis!**



Das Fachmagazin für .NET-Entwickler

Testen Sie jetzt 2 kostenlose Ausgaben und erhalten Sie unseren exklusiven Newsletter gratis dazu.

www.dotnetpro.de/probeabo

AIT

HoloLens auf der DWX

Seit April liefert Microsoft in den USA die Prototypen der HoloLens an handverlesene Software-Entwickler aus. Als eines der ersten Unternehmen in Europa hat die Stuttgarter AIT GmbH & Co. KG zwei Exemplare erhalten. Diese begeisterten auf der DWX in Nürnberg die Teilnehmer. In der Ausstellung schwebte eine dreidimensionale Galaxie und an allen Ecken und Enden tummelten sich virtuelle Tiere und Fabelwesen. So zumindest stellte es sich für die Besucher dar, die sich am Stand der AIT zum ersten Mal die Microsoft HoloLens aufsetzen konnten. Für die Außenstehenden



AIT präsentierte auf der Developer Week (DWX) in Nürnberg erstmalig die Microsoft HoloLens in Deutschland

sah es so aus, als seien die Träger plötzlich in eine andere Welt teleportiert.

Die Mixed-Reality-Brille HoloLens ist der erste holografische Computer. Im Gegensatz zu anderen Virtual-Reality-Brillen kann sie unabhängig von einer Verbindung zu einem Rechner oder Smartphone absolut selbstständig arbeiten, da in ihr ein vollständiger Computer mit einer holografischen Recheneinheit integriert ist. www.aitgmbh.de

Die Zukunft der Apps**Inno{Hacks}: Die Apps der Gewinner**

An einem Wochenende Anfang Juni kamen in der Dualen Hochschule Baden-Württemberg Karlsruhe 75 Entwickler, Designer und Innovationsexperten beim Hackathon Inno{Hacks} zusammen. In Teams von bis zu fünf Personen wurden in-



Anfang Juni fand in der Dualen Hochschule Baden-Württemberg Karlsruhe das Hackathon Inno{Hacks} statt

nerhalb von 24 Stunden neue Ideen für Apps entworfen und erste Prototypen entwickelt.

Innerhalb von drei Minuten galt es die Jury zu überzeugen, um in einer von drei Preiskategorien den ersten Platz beanspruchen zu können.

Die Preise waren mit jeweils 500 Euro honoriert und wurden von der Stadt Karlsruhe, vertreten durch Herrn Peter Behringer und Erik Wohlfeil, an die Teilnehmer vergeben.

Inno{Hacks} war ein Teil von Hack Germany, einem Hackathon-Wochenende in Deutschland.

Über 150 Teilnehmer kamen Berlin, München und Karlsruhe zusammen, um gemeinsam neue Ideen zu entwickeln. Veranstalter des Hackathons war Hackerstolz e. V. (www.hackerstolz.de/de) ein Verein zur Förderung der digitalen Kultur in Deutschland.

Dominik Doerner, Mitorganisator des Hackathons: »Die Atmosphäre am Hackathon war super. Viele Leute, viele kreative Ideen. Wir sind gespannt, ob wir eine der Ideen in einem Start-up wiedersehen!« Manuel Rauber, Mitorganisator, ergänzt: »Dank der Mensa waren die Teilnehmer mit leckerem Essen und kühlen Getränken versorgt. So konnten sie sich vollkommen auf ihre Ideen konzentrieren. Wir freuen uns, nächstes Jahr einen weiteren Hackathon an der DHBW Karlsruhe ausrichten zu können.«

<http://hackgermany.de>

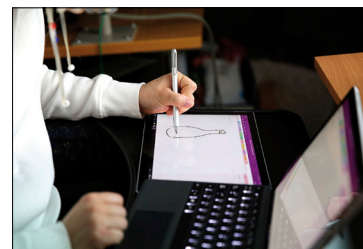
Norwegian University of Science and Technology**Handschrift schlägt Tastatureingabe**

Forschungsergebnisse der Norwegian University of Science and Technology (NTNU) in Trondheim ergaben: Lernende verarbeiten digitale Notizen mit dem Stift besser als über die Eingabe per Tastatur.

Die Eingabe von digitalen handschriftlichen Notizen ist ein effizienteres Mittel zur Aufnahme von Informationen als die Nutzung eines Laptops mit Tastatur. Zu diesem Ergebnis gelangen Wissenschaftler der Norwegian University of Science and Technology (NTNU) in Trondheim in einer neuen Studie zur Hirnforschung. 2-in-1-Geräte wie das Microsoft Surface Pro 4 oder Surface Book rücken damit weiter in den Fokus von Lehrkräften und Dozenten, denn sie ermöglichen neben der Bedienung via Tastatur und Maus zusätzlich Touch-Eingaben.

Angelehnt an vorangegangene Studien, die darauf schließen lassen, dass Informationen, die handschriftlich notiert werden, in tieferen Arealen des Gehirns verarbeitet werden, führten die Professoren Audrey van der Meer und Ruud van der Weel von der NTNU ein zweimonatiges Forschungsprojekt mit Studenten durch. Im Rahmen der Studie wurden 20 Studenten gebeten, drei verschiedene Aufgaben (Tippen, Schreiben, Zeichnen) zu erledigen. Die Studenten verwendeten dabei ein Surface Pro 4 und trugen eine mit über 250 Sensoren bestückte Kopfbedeckung, anhand derer die Hirnaktivitäten während der Studie aufgezeichnet wurden.

Die Ergebnisse belegen, dass die Studenten für die Zeichnungen mit einem Stift andere Bereiche des



Stift oder Tastatur: Stift schlägt bei der Verarbeitung digitaler Notizen die Eingabe per Tastatur

Gehirns nutzen als für die Eingabe ihrer Notizen mit der Tastatur des 2-in-1-Geräts. So zeigten die Untersuchungen, dass bei den Studenten tiefer liegende Areale im Gehirn aktiv waren, wenn sie handschriftliche und visuelle Notizen anfertigten. Aus vergleichbaren vorausgegangenen Studien ist bekannt, dass sich die Aktivierung der tiefer liegenden Hirnareale positiv auf das Lernen auswirkt.

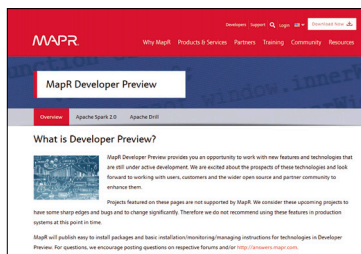
Die Wissenschaftler geben nun die Empfehlung, weltweit in Klassenzimmern und Hörsälen wieder auf handschriftliche Aufzeichnungen mit dem Stift zu setzen.

www.ntnu.no

MapR

Neue Apache-Spark-Distribution

MapR Technologies Inc. erweitert sein Angebot um eine Apache-Spark-Distribution. Die ab sofort verfügbare Distribution für den Einsatz



MapR vereint die Stärken von Apache Spark und seiner MapR Converged Data Platform

im Unternehmen soll die Fähigkeiten des Frameworks mit den Features der MapR-Plattform kombinieren und so dessen Benutzung einfacher und verlässlicher machen. Mit dem neuen Produkt will MapR dazu beitragen, die weitere Verbreitung von Spark in der Big-Data-Welt zu erleichtern.

In der neuen Distribution enthalten ist die neueste Version von Adobe Spark. Features wie In-Memory-Processing für Big Data ermöglichen die schnellere Entwicklung von Anwendungen und die Wiederverwendung von Code über Batch-, Interactive- und Streaming-Anwendungen hinweg.

MapR wird die Distribution auch als Quick Start Solution anbieten, die vorge-

fertigte Templates sowie Konfiguration und Installation umfasst. Zu den üblichen Einsatzbereichen von Spark gehören der Aufbau von Datenpipelines und die Entwicklung von Analyse-Anwendungen, die auf maschinellem Lernen basieren.

Die Spark-Distribution erfüllt auch anspruchsvolle Aufgaben, darunter erweiterte Analysen wie Batch-Processing, maschinelles Lernen, prozedurales SQL und Graph Computing. Spark profitiert dabei von den patentierten Features der MapR-Plattform. Web-Scale-Storage, hohe Verfügbarkeit, Mirroring, Snapshots, NFS, Integrated Security und Global Namespace erleichtern die Benutzung im Unternehmenskontext und sorgen für mehr Verlässlichkeit.

Durch die native Integration entsteht die vielleicht einzige verlässliche Plattform für Spark-Workloads. Erweitert werden kann die Distribution um Echtzeit-Streaming und Operational Analytics – mit MapR-Streams, MapR-DB und Hadoop als möglichen Add-ons.

MapR Platform including Spark ist im Rahmen der MapR Converged Community Edition und der MapR Converged Enterprise Edition verfügbar.

www.mapr.com

Fraunhofer-Allianz Big Data Fraunhofer gründet Big Data Business Club

Big Data ist bei vielen deutschen Unternehmen in der Führungsebene angekommen. Um Entscheidern fachliche Orientierung zu bieten und den Austausch auf strategischer Ebene

Bitkom Research

83 Prozent der Unternehmen nutzen Outsourcing

Laut einer Untersuchung des Bundesverbands Informationswirtschaft, Telekommunikation und neue Medien e.V. (Bitkom) haben mehr als acht von zehn Unternehmen ihre IT-Leistungen ausgelagert. Transport- und Logistikunternehmen beauftragen am häufigsten externe IT-Dienstleister.

Die große Mehrheit der Unternehmen in Deutschland setzt bei ihrer IT auf externe Dienstleister. Das hat eine repräsentative Befragung im Auftrag des Digitalverbands Bitkom ergeben. Demnach haben 82 Prozent der befragten Unternehmen Teile ihrer IT-Leistungen ausgelagert.

Am häufigsten werden externe IT-Dienstleister von Unternehmen aus der Transport- und Logistikbranche (86 Prozent) sowie dem Handel (83 Prozent) in Anspruch genommen. Am seltensten greifen Ver- und Entsorger (76 Prozent) sowie Banken und Finanzdienstleister (77 Prozent) auf IT-Outsourcing zurück.

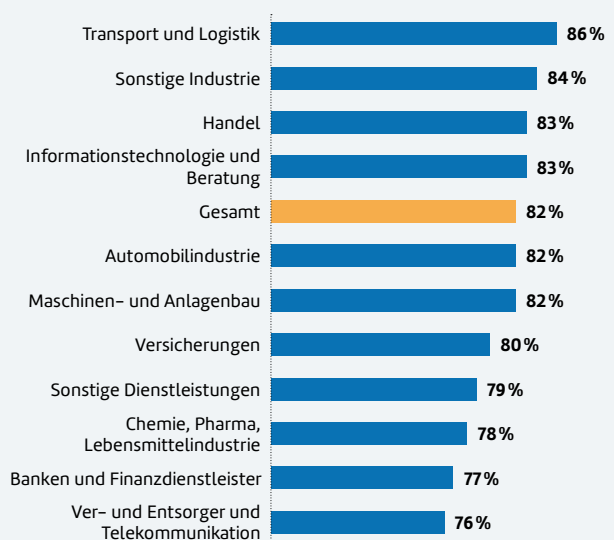
»Für die Wartung der Unternehmens-IT oder zur Lösung alltäglicher Computerprobleme benötigen nur wenige Unternehmen eigene IT-Experten«, sagt Bitkom-Experte Frank Früh. Auch sind Anbieter von IT-Leistungen oft breit aufgestellt und bieten neben einfachem IT-Anwendersupport zum Beispiel Cloud-Lösungen oder IT-Sicherheitsstrategien an.

»Externe Dienstleister ermöglichen insbesondere kleineren Unternehmen den Zugriff auf technische Lösungen, deren Entwicklung und Instandhaltung in der Regel weit über die Möglichkeiten einer unternehmenseigenen IT-Fachkraft hinausgeht«, so Früh.

Die Umfrageergebnisse stammen aus dem Bitkom Digital Office Index 2016. Er zeigt den aktuellen Stand und die Effekte der Digitalisierung deutscher Unternehmen.

www.bitkom.org/doi

Hat Ihr Unternehmen IT-Leistungen an einen externen IT-Dienstleister ausgelagert?



Basis: Unternehmen ab 20 Mitarbeiter (n=1108); Antwortmöglichkeit »ja« in Prozent

web & mobile developer 9/2016

Quelle: Bitkom Research

Developer-Week-2016-Umfrage

Ohne Risiko, keine Innovationen

Deutschland braucht einen lockeren Umgang mit dem Thema »Scheitern«. Das ist das Ergebnis einer deutschlandweiten Studie und gleichzeitig der Appell von über 200 IT-Professionals an die deutsche Gesellschaft. Misserfolg sollte stattdessen als Teil des Lernprozesses akzeptiert werden, der untrennbar mit Fortschritt und Wachstum verknüpft ist.

Dementsprechend sehen fast 80 Prozent der IT-Professionals die Angst vor dem Scheitern als Ursache dafür, dass deutsche Gründer im Vergleich zu vielen anderen Ländern weit weniger innovationsfreudig sind. Durchgeführt wurde die Umfrage von der Developer Week, einer der größten unabhängigen Entwickler-Konferenzen in Europa, die vom 20. bis 23. Juni 2016 in Nürnberg stattfand. Das Wort »Scheitern« ist im deutschen Sprachgebrauch sehr negativ besetzt und spiegelt die in diesem Land vorherrschende Angst vor dem Misserfolg wider. Dem zum Trotz haben aber IT-Professionals mehr Vertrauen zu Personen, die um die Erfahrung eines Misserfolgs »reicher« sind, als zu Personen, die diese Erfahrung nicht aufzuweisen haben.

www.developer-week.de

Frage: Deutschen wird oft vorgeworfen, gescheiterten Personen keine zweite Chance zu geben. Wie denken Sie darüber?

1 x Scheitern ist okay, beim zweiten Mal hat die Person verspielt.



Scheitern ist nur dann verwerflich, wenn es durch fahrlässiges Verhalten verursacht wurde.



Scheitern ist nichts Schlimmes.



Gescheiterten Personen kann man nicht mehr vertrauen.



Scheitern ist ein lebenslanges Stigma.



Man muss gescheiterten Personen immer eine zweite Chance geben.



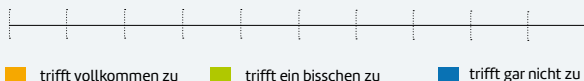
Scheitern ist dann okay, wenn man zu seinen Fehlern steht und nichts vertuschen will.



Scheitern ist Bestandteil des Innovationsprozesses.



Man muss Misserfolge beziehungsweise Scheitern thematisieren.



web & mobile developer 9/2016

Quelle: Developer Week

zu fördern, hat die Fraunhofer-Allianz Big Data eine bundesweite Plattform zur branchenübergreifenden Vernetzung und Zusammenarbeit geschaffen.

Der Big Data Business Club fördert den Austausch zwischen Wirtschaft und Wissenschaft und will auf diese Weise ein innovationsfreundliches Klima in der deutschen IT-Landschaft schaffen.

Darüber hinaus stehen die Institute der Fraunhofer-Allianz den beteiligten Unternehmen bei der strategischen Planung und Umsetzung ihrer Big-Data-Projekte zur Seite. Schirmherr des Big Data Business Clubs ist Prof. Dr. Reimund Neugebauer, Präsident der Fraunhofer-Gesellschaft.

In vielen Branchen sind Big-Data-Anwendungen bereits fest in die Unternehmensabläufe integriert. Damit die neuen Datentechnologien ihr volles Potenzial entfalten können, beziehen viele Firmen sie in die strategische Unternehmensplanung mit ein.

IT-Entscheider stehen dabei vor der Herausforderung, sich in der noch weitgehend unstrukturierten Big-Data-Landschaft zu orientieren, passende Technologien zu finden und in konkrete Anwendungen zu überführen.

Ein optimaler Transfer von neuesten Forschungsergebnissen in die Praxis kann nur erfolgen, wenn sich Entscheider und Forscher im unmittelbaren Austausch befinden. Aus diesem Grund hat die Fraunhofer-Allianz Big Data den Big Data Business Club gegründet. IT-Führungskräften und CDOs bietet er eine exklusive Plattform, um mit Entschei-

dern aus allen Branchen sowie renommierten Big-Data-Forschern in Kontakt zu treten und sich bei der Umsetzung konkreter Projekte unterstützen zu lassen.

Neben den Möglichkeiten zur Vernetzung bietet der Club ein umfassendes Leistungspaket. So erhält jedes Unternehmen einen persönlichen Ansprechpartner aus



Prof. Dr. Reimund Neugebauer, Präsident der Fraunhofer-Gesellschaft und Schirmherr des Big Data Business Clubs

der Fraunhofer-Allianz Big Data, der ihm bei fachlichen Fragen und Ideen für konkrete Projekte zur Seite steht.

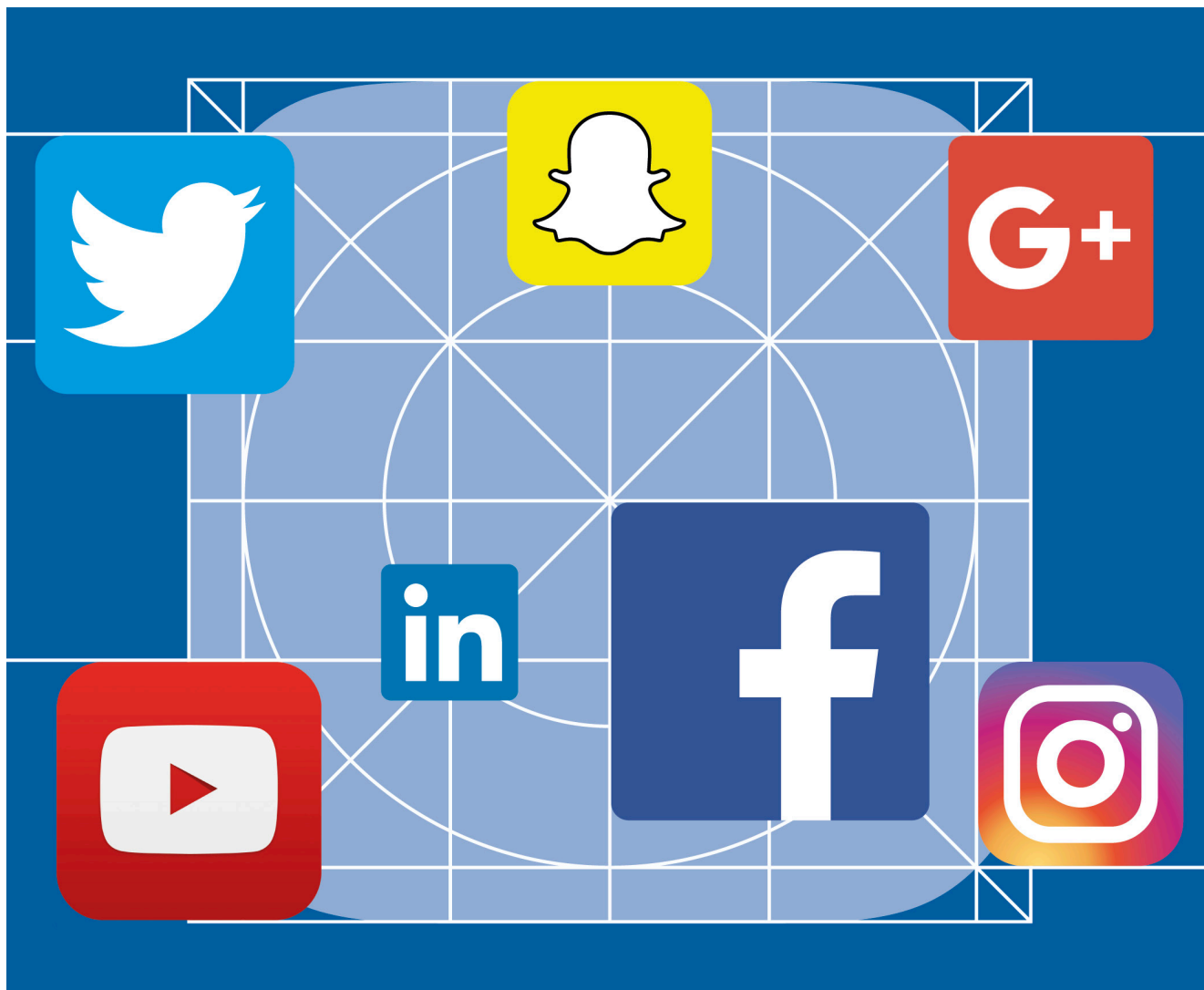
Zudem beraten die beteiligten Fraunhofer-Institute zur strategischen Projektplanung, liefern Technologiescreenings und Best-Practice-Anwendungen und vermitteln Kontakte zu Schlüsselpersonen aus Wissenschaft, Politik und Wirtschaft. Mit ihrem Big-Data-Schulungsprogramm verfügt die Fraunhofer-Allianz Big Data außerdem über weitreichende Erfahrung bei der Ausbildung und Rekrutierung von Big-Data-Spezialisten und steht den beteiligten Unternehmen auch hierbei beratend zur Seite. www.bigdata-business-club.de



Developer Week 2017

26.-29. Juni 2017,
Messe Nürnberg





DEEP INTEGRATION MIT SOZIALEN NETZEN

Reizvolle Features

Soziale Netze bieten Entwicklern inzwischen eine Fülle an überaus reizvollen Features.

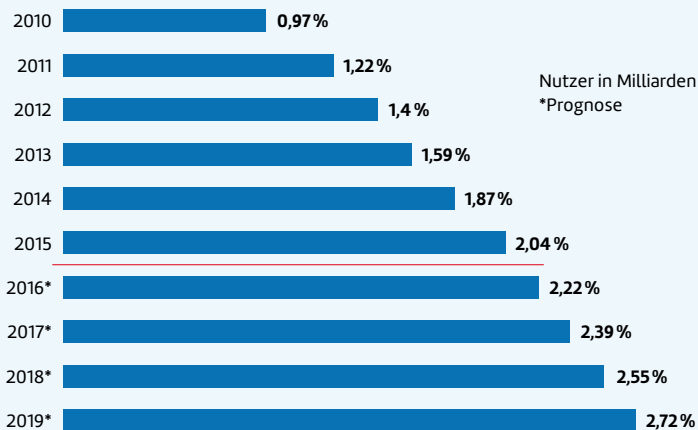
Auch den Technik-Skeptikern unter den Endverbraucher sind soziale Netze längst kein Fremdwort mehr. Das Marktforschungsinstitut Statista schätzt die Anzahl aktiver Nutzer sozialer Medien weltweit auf über 2 Milliarden ein (Bild 1). Alleine der Platzhirsch Facebook zählte im vergangenen Jahr weltweit knapp 1,5 Milliarden monatlich aktiver Nutzer. Das soziale Netzwerk verarbeitet im Durchschnitt 890 Milliarden API-Aufrufe pro Tag.

Fest im Sattel: Facebook

Mit einem Marktanteil von knapp über 85 Prozent aller Seitenabrufe weltweit führt Facebook, gefolgt von Pinterest mit 6,6 Prozent und Twitter mit 4,8 Prozent, berichtete Statista im Februar 2016.

Auch in Deutschland liegt der weltweite Platzhirsch mit etwa 23,5 bis 28 Millionen monatlich aktiver Mitglieder – umgerechnet jedem zweiten Internetnutzer – eindeutig auf Platz eins. 19 Millionen dieser Mitglieder kehren laut Facebook jeden Tag wieder; dies entspricht etwa der Reichweite der drei größten Tageszeitungen Deutschlands zusammengenommen (Media Impact). Jeden Monat sind mehr als 18 Millionen Deutsche – also umgerechnet nahezu sieben von zehn Smartphonebesitzern – mit einem Mobilgerät auf Facebook unterwegs; nahezu drei von vier kommen täglich wieder. Insgesamt hat die deutsche Facebook-Gemeinde die Marke 600 Millionen Seitenaufrufe pro Monat im Dezember 2014 durchbrochen. Google+ und Twitter brachten es mit je circa 40 Millionen Besuchen auf einen geteilten Platz zwei.

Nutzer von sozialen Netzwerken



Anzahl der Nutzer von sozialen Netzen weltweit in Milliarden (2010–2015) und eine Prognose (2016–2019) (Bild 1)

web & mobile developer 9/2016

Quelle: Statista

Auch bei der Nachrichtenverbreitung dominiert in Deutschland nicht der Mikroblogging-Dienst Twitter, sondern eben Facebook, und zwar mit einem erdrückenden Marktanteil von über 90 Prozent. Dies ergab im Februar 2015 eine Untersuchung durch Forscher aus dem Fachbereich Informatik der Technischen Universität Darmstadt (Bild 2).

Pinterest führt dagegen deutlich im Hinblick auf das weltweite Wachstum der aktiven Benutzeraccounts. Zwischen dem 1. bis 2. Quartal 2014 und dem 4. Quartal 2014 bis 1. Quartal 2015 konnte Pinterest laut dem Statistikportal Statista gemessen an der Zahl aktiver Nutzer um stolze 97 Prozent zulegen und bringt es auf 100 Millionen aktiver Nutzer weltweit. Im selben Zeitraum ist Facebooks eigene Pinterest-Alternative Instagram mit ihren 400 Millionen aktiven Nutzern weltweit (davon 9 Millionen in Deutschland) um nur 46 Prozent gewachsen, während das Facebook-Netzwerk selbst um 8 Prozent schrumpfte und LinkedIn um 39 Prozent zulegte.

YouTube-Marktanteil

Das soziale Netzwerk Google+ floppte ungeachtet Googles dominierender Position bei der Internetsuche und des Erfolgs der Android-Plattform. Dennoch führt Googles Videoplattform YouTube unter anderem in Deutschland mit einem nach wie vor beachtlichen Marktanteil von über 50 Prozent (Bild 3).

Neben sozialen Netzwerken mit ausgeprägter globaler Reichweite (Facebook, Twitter, LinkedIn et cetera) haben auch Netzwerke mit rein europäischem oder rein deutschem Schwerpunkt eine gewisse Popularität erreicht. Das in Hamburg beheimatete Netzwerk Xing bringt es im

DACH-Raum eigenen Aussagen zufolge auf eine Reichweite von über 9,2 Millionen Menschen (davon 869.000 zahlende Mitglieder) und auf über 171 Millionen Seitenaufrufe pro Monat, laut IVW mehr als alle seine Mitbewerber unter den Geschäftsportalen zusammengenommen.

Die anhaltende Popularität dieser Dienste verspricht eine langfristige Beständigkeit und hat eine enorme Anziehungskraft für Unternehmen. Aufgrund ihrer beachtlichen Reichweite haben sich soziale Netze in vielseitig nutzbare Marketing-Plattformen verwandelt.

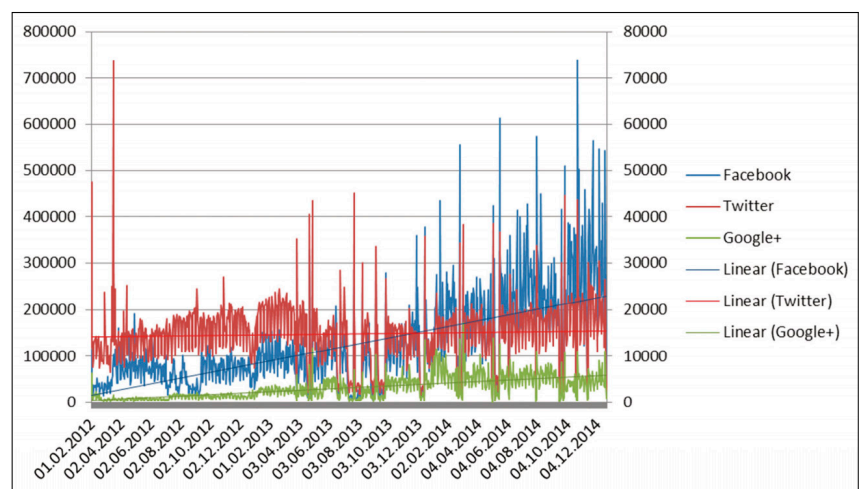
Auf der einen Seite fungieren diese Dienste als ein Werbemedium im klassischen Sinne, auf der anderen Seite bieten sie Instrumente der Marktforschung, deren unübertroffene Genauigkeit und Kundennähe den Unternehmen einen deutlichen Mehrwert bietet. Dies spiegelt sich auch in den Benutzerzahlen wider. Drei von vier weltweiten Großunternehmen aus der Fortune-500-Liste setzen laut Statista das soziale Netzwerk Twitter ein.

Facebooks eigenen Aussagen zufolge gibt es weltweit mehr als 50 Millionen Unternehmens- oder Markenwebseiten. Über 9 Millionen mobiler Apps und Webanwendungen verfügen über eine Integration mit Facebook.

Vorteile der Integration

Für die Entwickler mobiler Apps und Webapplikationen verspricht eine derart hohe Popularität und Marktdurchdringung sozialer Plattformen einen potenziell sehr hohen Nutzen einer durchdachten Integration. Die Möglichkeiten beinhalten eine Vielzahl von Ansätzen, von einer sicheren Benutzer-Authentifizierung über den Zugriff auf Benutzerdaten zur Bereitstellung erweiterter App-Funktionalität bis hin zur Akquisition neuer Anwender.

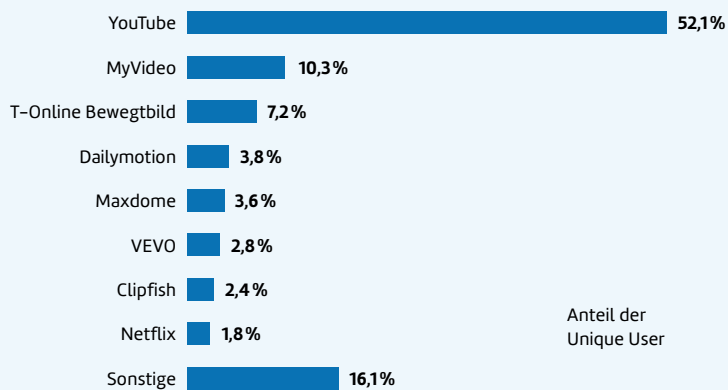
Die erfolgreichsten sozialen Netze sind nicht nur im gesellschaftlichen Kontext sozialer Beziehungen, sondern auch ►



Engagiert: Die Anzahl von Gefällt-mir-Interaktionen, Tweets und +1-Klicks in Deutschland pro Tag im untersuchten Zeitraum weist auf eine klare Dominanz von Facebook hin (Bild 2)

Quelle: Technische Universität Darmstadt

Nutzung von Videoplattformen



Eindeutig: Die Verteilung der Marktanteile der Videoplattformen in Deutschland zeigt eine klare Dominanz von Googles YouTube (Bild 3)

web & mobile developer 9/2016

Quelle: Statista

auf der API-Ebene äußerst integrativ. Gut dokumentierte APIs und die oft kostenlose Abwicklung von API-Zugriffen haben die Verbreitung dieser Dienste noch weiter erhöht. Die Kunst besteht allerdings nicht darin, die hundertsiebensiebzigste Gefällt-mir-Schaltfläche in eine App zu integrieren, sondern eine Implementierung zu entwickeln, die in die Tiefe der Möglichkeiten sozialer Netze greift und einen konkreten Nutzen daraus zieht. Soziale Netzwerke fungieren dann nicht bloß als Empfänger von Daten, sondern auch als Anbieter von Softwarediensten und Daten aus der Cloud.

Facebook unterscheidet fünf unterstützte Funktionalitätsbereiche für externe Software-Anwendungen:

- Benutzer-Authentifizierung via Facebook-Login,
- Veröffentlichen von Benutzerdaten aus externen Anwendungen auf Facebook,
- Facebook-Analytics für mobile Apps,
- Monetarisierung mobiler Apps mittels Facebook,



Virtuell: In dem Spiel Overwatch können Benutzer mit ihren Freunden aus der Facebook-Gemeinde interagieren (Bild 4)

- Bots für die kürzlich vorgestellte Facebook-Messenger-Plattform.

Nicht jedes soziale Netzwerk steht jeder Art von Integration und jeder Art von Anwendung in gleichem Maß aufgeschlossen gegenüber. Facebook setzt beispielsweise verstärkt auf Spiele. Facebooks eigener Marktplatz für Anwendungen mit Facebook-Integration ist unter der Adresse <https://www.facebook.com/games> verfügbar.

Je nach Art der Anwendung und der Zielgruppe mag Facebook nicht automatisch die beste Wahl darstellen. Vor der Anbindung einer Website oder einer mobilen App an soziale Medien empfiehlt es sich, ein klares Ziel zu definieren und die geeignete Plattform zu wählen.

Webanwendungen und mobile Apps können aus den Lösungen, welche die Anbieter sozialer Medien entwickelt haben, dank einer durchdacht implementierten tiefen Integration einen sehr hohen Nutzen ziehen.

Hohe Reichweite

Die führenden sozialen Netze haben nicht nur eine hohe Reichweite erreicht, sie haben sich inzwischen in leistungsstarke Web-Entwicklungsplattformen verwandelt. Sie verfügen nicht nur über massive Datenbestände rund um das Privatleben ihrer Endbenutzer (was den Datenschützern ja ernsthafte Sorgen bereitet), sondern können diese Rohdaten auswerten und die Resultate als konkret umsetzbare Handlungsempfehlungen oder Marketing-Features ihrer Plattformen nach außen hin bereitstellen.

Für die Entwickler von Webanwendungen und mobilen Apps gibt es zahlreiche gute Gründe, die Einbindung der bereitgestellten APIs geeigneter Dienste ernsthaft in Erwägung

Listing 1: Artikel in Facebook Open Graph

```
<html>
<head prefix="og: http://ogp.me/ns#
fb: http://ogp.me/ns/fb#
article: http://ogp.me/ns/article#">

<meta property="fb:app_id"
content="218565732440027" />
<meta property="og:type" content="article" />
<meta property="og:url"
content="http://www.webundmobile.de/update/
datenbank/schneller-dokumentenspeicher-1105518.
html" />
<meta property="og:title"
content="Schneller Dokumentenspeicher" />
<meta property="og:image"
content="http://www.webundmobile.de/img/6/2/8/5/
4/2/raptorDB_w600_h445.jpg" />
```

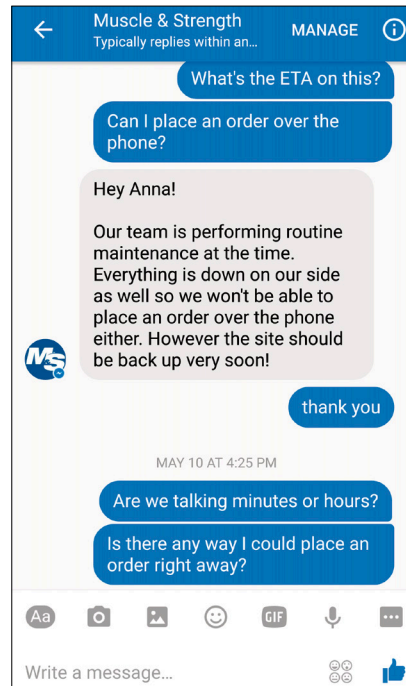

zu ziehen, um den eigenen Endanwendern neue Features bereitzustellen und eine verbesserte Benutzererfahrung zu bieten.

Das jüngste Beispiel für die Vorteile einer engen Facebook-Integration ist das Facebook-Login und die Facebook-Live-Video-Unterstützung durch das Overwatch-Game von Blizzard Entertainment und durch die Battle.net-Online-Spielgemeinde (Bild 4).

Offenbar scheint dieses Facebook-Feature äußerst populär zu sein, da sich immerhin sieben Millionen Fans des Spiels allein in der ersten Woche der Verfügbarkeit dafür entschieden haben.

Die Idee, Video-Live-Streaming in Applikationen und soziale Netze einzubinden, ist sicherlich nicht neu, denn immerhin tritt Facebook damit in direkte Konkurrenz zu Amazons Twitch und Googles YouTube, die bei der Unterstützung von Live-Videostreaming ebenfalls um die Gunst von Web- und Mobil-App-Entwicklern buhlen.

Ob es darum geht, die Benutzer-Authentifizierung zu vereinfachen, ohne die Sicherheit aufs Spiel zu setzen, oder darum, den Endanwendern einen komfortablen Zugriff auf ihre eigenen Fotos und andere Daten zu ermöglichen, stets gilt: Die Anbindung an soziale Netze benötigt leistungsstarke, langfristig unterstützte APIs. Eine Deep-Integration-Anbindung von Webanwendungen und



Ein Draht zum Kunden: Facebooks Chat-App Messenger findet auch im Kundendienst Anwendung; sie verschafft den Unternehmen Kundennähe (Bild 5)

mobilen Apps an soziale Netze wie Facebook hat viele mögliche Anwendungsgebiete. Eine offensichtliche Anwendungsmöglichkeit für Messenger-Bots wäre beispielsweise ein Facebook-gestützter Kundendienst für Unternehmen.

Ob Sie eine Pizza bestellen, ein Taxi rufen oder ein Paket nachverfolgen wollen: Bei klar definierten Aufgaben sind Messenger-Bots für Facebook geradezu ideal. Doch es handelt sich bei Weitem nicht um das einzige Anwendungsgebiet (Bild 5).

Bei allen Anwendungen, welche die Endverbraucher adressieren, empfiehlt sich eine Integration mit Facebook: So können Sie Ihre Anwender dort treffen, wo sie ohnehin verweilen. Ein konkretes Beispiel einer gelungenen Integration stellt die Anbindung von Skype an Facebook dar. Soweit es Skype betrifft, hat Microsoft ganze Arbeit geleistet.

Das Netz der kontextuellen Beziehungen

Aus der Sicht der Softwareprogrammierung stellt ein soziales Netz eine überaus reizvolle Plattform dar. Das Grundgerüst

eines sozialen Netzwerks bildet ein sogenannter Social Graph, ein Modell der Beziehungen seiner Mitglieder untereinander, gegebenenfalls unter Berücksichtigung von Inhalten, mit denen diese Benutzer interagiert haben.

Ein Benutzer mag ein Update in seiner Timeline posten, ein anderer kommentiert es vielleicht, der Nächste wiederum klickt auf den Gefällt-mir-Button und noch jemand anderer erkennt einen Freund in dem Foto und taggt es mit seinem Namen. Diese Aktivitäten reflektieren Zusammenhänge zwischen Menschen, Medien und geografischen Standorten zu jedem gegebenen Zeitpunkt.

Durch die Analyse dieser Zusammenhänge kann das soziale Netzwerk den betreffenden Inhalten ihre Bedeutung in Relation zu den betroffenen Menschen entnehmen. So wird eine Wissensbasis über Zielgruppen und Marktnischen entwickelt und eine neuartige Suchfunktionalität ermöglicht – die sogenannte Graph Search.

Graph Search erlaubt es, mit intelligenten, kontextabhängigen Suchanfragen fein granulierte Informationen aus der sozialen Plattform zu extrahieren. Facebooks Graph Search kommt unter anderem für die Marktforschung wie gerufen: Analysesoftware kann die Präferenzen der anvisierten Zielgruppe erforschen und demografische Kriterien mit persönlichen Vorlieben im Kontext sozialer Interaktionen untersuchen. Die Ära fein granulierter Marktforschung auf der Basis von Big Data aus sozialen Netzen ist offiziell angebrochen.

Das Open-Graph-Protokoll

Die Anbindung von Webanwendungen und mobilen Apps in Facebooks Social Graph erfolgt mit Hilfe des Open-Graph- ►

Listing 2: API-Aufruf von Facebook in PHP

```
$request = new FacebookRequest(
    $session,
    'POST',
    '/me/objects/article',
    array (
        'object' => '{"fb:app_id":"218565732440027",
        "og:type":"article","og:url":
        "http://www.webundmobile.de/update/
        datenbank/schneller-
        dokumentenspeicher-1105518.html","og:title":
        "Schneller Dokumentenspeicher","og:image":
        "http://www.webundmobile.de/
        img/6/2/8/5/4/2/raptorDB_w600_
        h445.jpg"}',
    )
);

$response = $request->execute();
$graphObject = $response->getGraphObject();
```

Listing 3: API-Aufruf von Facebook mit Android SDK

```
Bundle params = new Bundle();
params.putString("object", "{\"fb:app_id\": \"218565732440027\", \"og:type\": \"article\", \"og:url\": \"http://www.webundmobile.de/update/datenbank/schneller-dokumentenspeicher-1105518.html\", \"og:title\": \"Schneller Dokumentenspeicher\", \"og:image\": \"http://www.webundmobile.de/img/6/2/8/5/4/2/raptorDB_w600_h445.jpg\"}");

new GraphRequest(
    AccessToken.getCurrentAccessToken(),
    "/me/objects/article",
    params,
    HttpMethod.POST,
    new GraphRequest.Callback() {
        public void onCompleted(
            (GraphResponse response) {
                /* das Resultat nutzen */
            }
        )
    }
).executeAsync();
```

Protokolls. Es ermöglicht die Integration von Benutzeraktivitäten außerhalb von Facebook mit dem sozialen Graphen der Facebook-Plattform. Das Rückgrat der Anbindung bildet das Graph API von Facebook (aktuelle Version 2.6).

Um Elemente Ihrer Webanwendung oder mobilen App in Facebooks Open-Graph-Modell zu integrieren, müssen Sie diese mit korrekten Tags versehen. Ein Beispiel-Markup für einen Blog-Post oder einen Artikel in Facebook Open Graph finden Sie im Listing 1. Den zugehörigen API-Aufruf in PHP illustriert Listing 2.

iOS-Entwicklern stehen die API-Aufrufe FBSDKGraphRequest und FBSDKShareAPI mit dem Open-Graph-Modell FBSDKShareOpenGraphContent zur Verfügung. Listing 3 erstellt das benötigte Markup unter Verwendung des Android SDK.

Mit dem Open-Graph-Debugger von Facebook (<https://developers.facebook.com/tools/debug>) können Sie überprüfen, wie sich Ihre Webseite präsentiert, wenn sie auf Facebook empfohlen wird (Bild 6).

Ähnliches gilt für LinkedIn und andere soziale Netze: Open-Graph-Tags sorgen dafür, dass der LinkedIn-Crawler beim Analysieren der Inhalte nicht raten muss. Nur so kann sichergestellt werden, dass der Inhalt der externen Webseite wie erwartet erfasst und auf LinkedIn korrekt dargestellt wird.

Anmeldung von Benutzern über soziale Netzwerke

Soziale Netzwerke bieten App-Entwicklern ausgereifte APIs zur Benutzer-Authentifizierung. Sie übernehmen dadurch die Rolle eines universellen Schlüssels, welcher dem Benut-

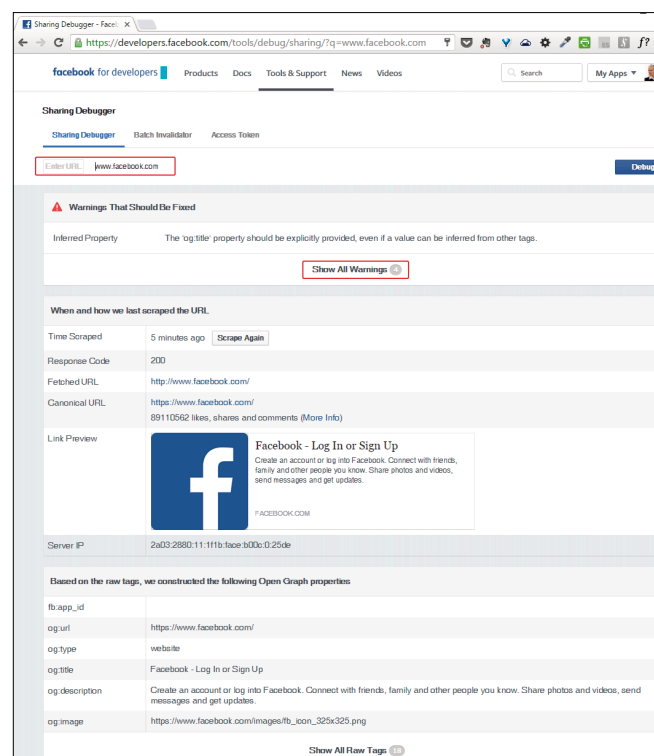
zer den Zugriff auf eine Vielzahl mobiler Apps und Webanwendungen gewährt. Der Komfort für den Endanwender ist dabei erheblich. Ob das Konzept der sozialen Authentifizierung aber so sicher ist, mag bezweifelt werden.

Ein Passwort-Hack an einer Stelle betrifft automatisch auch alle Apps, welche das betroffene soziale Netzwerk als universellen Authentifizierungsschlüssel verwenden. Hinzu kommt die Tatsache, dass viele Endbenutzer routinemäßig ein und dasselbe Passwort für mehrere soziale Netze verwenden. Eine richtig lange und komplizierte Zeichenkombination als Passwort mag ja hilfreich sein, um eine Brute-Force-Attacke auf den betroffenen Benutzeraccount zu verhindern, doch die Webanwendung kann beim Verdacht auf eine solche Attacke die Passwort-Überprüfung ohnehin verhindern.

Cyber-Sicherheit von sozialen Netzwerken

Mindestens genauso problematisch wie ein unsicheres Passwort ist die Unsitte, für alle Accounts ein und dasselbe Passwort zu verwenden, denn auch soziale Netze sind im Hinblick auf ihre Cyber-Sicherheit vor Hacks nicht ganz immun. Die Nachwirkungen des LinkedIn-Hacks von 2012 plagten die Betroffenen bis heute. Mark Zuckerbergs LinkedIn-Passwort wurde offenbar erst dieses Jahr auf dem schwarzen Markt verkauft – und funktionierte.

Die Hacker konnten sich damit nicht nur in Zuckerbergs LinkedIn-Account einloggen, sondern auch seine sozialen Identitäten von Pinterest und Twitter missbrauchen. LinkedIn hatte die Benutzerpasswörter als ungesalzene SHA-1-Hashes gesichert. Das Decodieren des Passworts von Zuckerberg war ein Kinderspiel.



Der Facebook-Debugger bescheinigt selbst der Facebook-Seite vier Open-Graph-Fehler (Bild 6)

Für konventionelle Passwörter scheint bald das letzte Stündlein geschlagen zu haben. Anbieter von Authentifizierungstechnologien, darunter Google und Amazon, experimentieren mit Alternativen zur passwortbasierten Authentifizierung. Apple Pay, Samsung Pay und Android Pay können ihre Benutzer mittels eines Fingerabdrucks authentifizieren.

Digits ist für Entwickler kostenfrei

Mit Digits (<https://get.digits.com>) startete Twitter einen entwicklerfreundlichen Dienst für passwortfreie Logins. Digits stellt dem Benutzer einen Anmeldedialog bereit. Hier trägt der Benutzer seine Telefonnummer ein und bekommt daraufhin eine SMS-Nachricht mit einem einmaligen Code, mit dem sich die Anmeldung abschließen lässt.

Für den Entwickler hat Digits mehrere Vorteile. Mobile Benutzer, die sich mit einer Telefonnummer authentifizieren, können mit Hilfe des Friends-Finding-Features Einträge aus ihrem Telefonbuch als Kontakte zu ihrer mobilen App hinzufügen und ihre Benutzerbasis erweitern. Digits rühmt sich einer Konversionsrate in Höhe von 85 Prozent. Der Dienst bietet Features wie Stimmverifikation (in 32 Sprachen), funktioniert in 217 Ländern und ist kostenfrei für Entwickler. Um Digits zu nutzen, benötigen Sie Twitters Fabric (<https://get.fabric.io>) mit dem Digits Kit von Twitter.

Facebook Social Plug-ins

Zur Integration von sozialen Netzen in eine Website können Sie eines der offiziellen Module des jeweiligen Anbieters oder einen der zahlreichen externen Dienste und Plug-ins verwenden.

Die offiziellen Facebook-Module, die sogenannten Social Plug-ins, finden Sie unter der Adresse http://developers.facebook.com/docs/plugins/?locale2=de_DE.

Um eines dieser Module in Ihre Website zu integrieren, wechseln Sie hier zur Detailseite der gewünschten Erweiterung und stellen die wichtigsten Parameter unter Verwendung des integrierten Widgets ein. So erhalten Sie den benötigten Code für mehrere alternative Methoden der Anbindung. Weitere Anpassungen des gewünschten Moduls können Sie anhand der Detailanleitung in einem Code-Editor Ihrer Wahl anpassen.

Die einfachsten Facebook-Plug-ins basieren auf einem iFrame, den Sie ganz einfach in Ihre Webseite einbinden; einige Module verwenden wiederum XFBML-Elemente (Extended Facebook Markup Language) und setzen das JavaScript SDK voraus. Bei diesen Modulen müssen Sie von Facebook eine App-ID (*appId*) erhalten, indem Sie Ihre Website als eine Facebook-Applikation registrieren. Hierzu navigieren Sie zur Adresse <https://developers.facebook.com/apps> und erstellen eine neue Facebook-App für das Web. Dabei handelt es sich um einen Eintrag Ihrer Website in der Facebook-Datenbank; Sie brauchen nicht wirklich eine Applikation zu entwickeln. Die so erzeugte App-ID müssen Sie dann in dem Code-Abschnitt der Webseite eintragen, der das JavaScript SDK initialisiert.

Das jähre Ende von Parse

Parse, Facebooks eigenes Backend für die standardisierte Bereitstellung von Serverdiensten an mobile Anwendungen, legte der Anbieter im Januar 2016 zum alten Eisen.

Bestehende Apps sollen bis Januar 2017 unverändert funktionieren, doch danach wird Facebook endgültig »den Stecker ziehen«. Entwickler, die ihre Apps auf Parse aufgebaut haben – alleine im Jahr 2014 waren es bereits eine halbe Million –, stehen daher bald vor einem Problem. Alternative Dienste, die Parse-ähnliche Funktionalität anbieten, beinhalten Amazon Web Services, Softlayer, Azure und Google App Engine.

Im Hinblick auf das Graph API verspricht Facebook neuerdings eine kontinuierliche Unterstützung einer jeden Version im Zeitrahmen von vollen zwei Jahren nach der erstmaligen Veröffentlichung.

Tipp: Mit dem neu vorgestellten API Upgrade Tool von Facebook können Sie die Auswirkungen eines Upgrades auf die Funktionsfähigkeit Ihrer Apps untersuchen.

Auf der diesjährigen F8-Konferenz im April folgte Facebook in Twitters Fußstapfen mit der Vorstellung einer Digits-Alternative namens AccountKit. Facebook AccountKit ermöglicht die Authentifizierung von Benutzern mobiler Apps und Webapplikationen ohne ein Passwort: anhand einer Telefonnummer oder einer E-Mail-Adresse (ein Facebook-Account ist nicht erforderlich).

Auch diese Methode der Authentifizierung ist nicht ohne ein gewisses Risiko: Wer irgendwann seine Telefonnummer aufgeben sollte, würde seine Identität möglicherweise an den neuen Besitzer verlieren. Doch auch diese Authentifizierungsmethode hat durchaus ihre Berechtigung, wenn sie ihrem Zweck angemessen ist und die Benutzererfahrung verbessert.

Sicherheitskritische Infrastruktur

Bei Apps zum Zugriff auf sicherheitskritische Infrastruktur oder sonstige relevante Backend-Systeme sollte man von sozialen Identitäten als dem universellen und einzigen Schlüssel zur Benutzer-Authentifizierung lieber Abstand nehmen. Für die meisten anderen mobilen Apps ist die Benutzer-Authentifizierung anhand von sozialen Identitäten in der Regel dennoch unbedenklich; dank der vereinfachten Registrierung reduziert sie die Absprungrate und erhöht somit die Reichweite Ihrer App.

Damit Ihre Apps und Webdienste auf persönliche Daten der Mitglieder eines sozialen Netzwerks zugreifen oder in seinem Auftrag Inhalte veröffentlichen können, müssen Sie von dem betroffenen Benutzer entsprechende Berechtigungen für Ihre App oder Ihre Webanwendung einholen. Der Vorgang setzt eine Authentifizierung des Endbenutzers gegenüber dem sozialen Netzwerk voraus. Soziale Netze und ähnliche Dienste – von Facebook über Twitter, LinkedIn bis hin zu GitHub – verwenden hierzu das OAuth-2.0-Protokoll ►

(<http://oauth.net>). Unter Verwendung von OAuth erteilt Ihnen der Endbenutzer seine Erlaubnis, ohne dabei sein Passwort preiszugeben.

Inhalte veröffentlichen am Beispiel von LinkedIn

Beim Einsatz des JavaScript-SDK von LinkedIn müssen Sie dieses erst einmal im `<head>`-Abschnitt der Webseite unter Angabe der natürlichen Sprache Ihres UI initialisieren (*lang: de_DE*). Die Schaltfläche zur Benutzer-Authentifizierung entsteht dann einfach mittels:

```
<script type="in/Login"></script>
```

Die Schaltfläche nutzt automatisch diejenige natürliche Sprache, die Sie zuvor initialisiert haben. Ein Klick auf diese Schaltfläche fördert einen Dialog mit der Aufforderung zur Authentifizierung in dem sozialen Netzwerk zutage. Der Vorgang selbst läuft asynchron ab; benötigt wird also eine Funktion, die das resultierende *auth*-Ereignis abfangen und auswerten kann. Diese Funktion binden Sie über das *onLoad*-Argument an das `<script>`-Tag der SDK-Initialisierung an, damit sie automatisch ausgeführt wird, sobald das SDK vollständig geladen ist.

Erst wenn diese Ereignisauswertungsfunktion ein *auth*-Ereignis empfangen hat, können Sie den generischen Wrapper für API-Aufrufe, *IN.API.Raw()*, einspannen, um mit Hilfe der REST-API-Schnittstelle Inhalte im Auftrag des betreffenden Benutzers zu veröffentlichen.

Wie sich das Ganze in JavaScript gestaltet, zeigt Listing 4. Da die Payload (also die Nutzlast) in diesem Beispiel in JSON erfasst wurde, müssen Sie in den `<head>`-Abschnitt Ihrer Webanwendung zusätzlich die folgenden HTTP-Header einfügen:

```
Content-Type: application/json
x-li-format: json
```

Beim Einsatz von XML gibt es diese Voraussetzung nicht.

Hat die Veröffentlichung geklappt, liefert LinkedIn den Statuscode HTTP 201 und folgende Informationen zurück:

```
{
  "updateKey": "UPDATE-1546982-35719543032...",
  "updateUrl": "https://www.linkedin.com/updates?discuss=...&scope=..."
}
```

Beim Wert von *updateKey* handelt es sich um eine eindeutige Kennung der Veröffentlichung, bei *updateUrl* um den direkten Link dazu. Sollte die App mehrfach dasselbe Update abschicken wollen, bekommt sie den Fehler HTTP 400 zurück.

Soziale Netze weisen jeder externen App typischerweise ein bestimmtes Kontingent an zulässigen API-Aufrufen kostenfrei zu. Das Kontingent ergibt sich in der Regel aus mehreren verschiedenen Beschränkungen. LinkedIn limitiert die Anzahl der API-Aufrufe pro App und zusätzlich noch pro Benutzer und pro Entwickler. Den aktuellen Stand Ihres Kontingents können Sie im Dashboard in Ihrem Entwickler-Account einsehen, bei LinkedIn beispielsweise auf der Übersichtsseite Ihrer App unter <https://www.linkedin.com/developer/apps>.

Facebook-Messenger-Plattform

Mit der Vorstellung der Messenger-Plattform für Chat-Bots (derzeit noch Beta) auf der F8-Konferenz in April 2016 sorgte Facebook für viel Aufsehen. ►

Listing 4: Inhalte auf LinkedIn veröffentlichen

```
<script type="text/javascript" src=
  "//platform.linkedin.com/in.js">
  api_key: IHR-API-SCHLUESSEL
  authorize: true
  onLoad: onLinkedInLoad
  lang: de_DE
</script>
<script type="text/javascript">
  function onLinkedInLoad() {
    IN.Event.on(IN, "auth", shareContent);
  }
  function onSuccess(data) {
    console.log(data);
  }
  function onError(error) {
    console.log(error);
  }
  function shareContent() {
    var payload = {
      "comment": "Besuchen Sie uns im Internet",
      "content": {
        "title": "web und mobile developer",
        "description": "Software entwickeln",
        "submitted-url": "http://webundmobile.de/",
        "submitted-image-url":
          "http://webundmobile.de/logo.png"
      },
      "visibility": {
        "code": „anyone“
      }
    };
    IN.API.Raw("/people/~shares?format=json")
      .method("POST")
      .body(JSON.stringify(payload))
      .result(onSuccess)
      .error(onError);
  }
</script>
```



.NET Developer Conference 2016

05. – 07. Dezember 2016 + Special am 08.12.2016
Köln, pullman Cologne

Tools + Technologien:

Softwarequalität, Frontend, Core, Any App

dotnet-developer-conference.de



SMART DATA Developer Conference

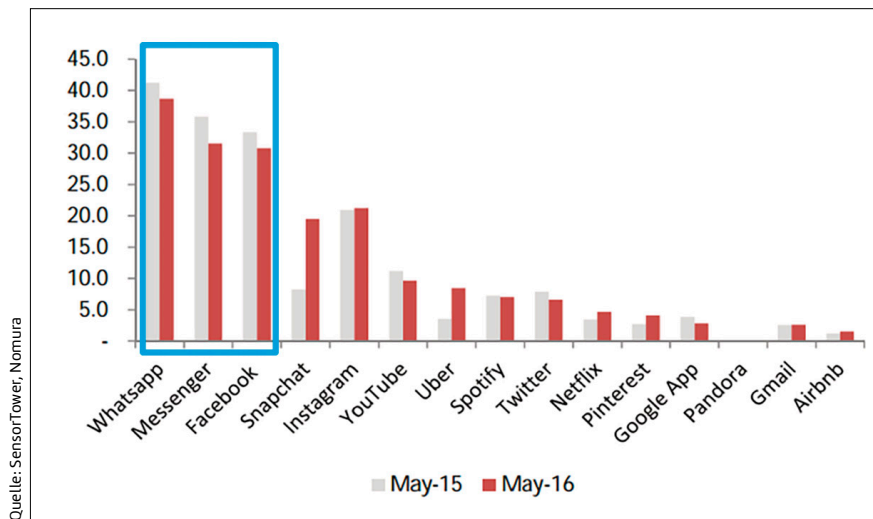
Big Data & Smart Analytics

06. Dezember 2016
Köln, pullman Cologne

Themen: Datenqualität, Visualisierung,
Analyse, Batch & Stream,
Processing, Tools & Frameworks

smart-data-developer.de





Dominant: Facebook führt im Hinblick auf die Gesamtanzahl weltweiter App-Downloads mit WhatsApp, Messenger und Facebook (Bild 7)

Die mobile App Facebook Messenger ist das Frontend für den gleichnamigen Chat-Dienst auf der Basis des quelloffenen MQTT-Protokolls (die Abkürzung steht für Message Queuing Telemetry Transport).

Facebook Messenger war laut Angaben des Anbieters im vergangenen Jahr (2015) die am schnellsten wachsende mobile App in den USA im Hinblick auf die Benutzerzahlen. Einer Studie von SensorTower und Nomura zufolge ist eine Trendwende allerdings nicht auszuschließen (Bild 7).

Facebook Messenger erreicht inzwischen 900 Millionen Anwender und dürfte damit mit WhatsApp ungefähr gleichgezogen haben. Der Dienst verarbeitet 60 Milliarden Nachrichten pro Tag, also ungefähr das Dreifache des weltweiten Volumens an SMS-Nachrichten (20 Milliarden).

Facebook-Benutzer interagieren tagtäglich mit mehreren Unternehmen. »Niemand möchte für jeden Dienstanbieter eine neue App installieren«, sagte Zuckerberg auf der F8-Konferenz. Die Facebook-Messenger-Plattform löst dieses Problem, indem sie Entwicklern ermöglicht, Bot-Apps für den Facebook Messenger zu schreiben.

Spracherkennung im Facebook Messenger

Die Facebook-Messenger-Plattform ist das Resultat der Übernahme des Start-ups Wit.ai, des Anbieters einer innovativen Spracherkennungsplattform für IoT (das Internet der Dinge), durch Facebook. Facebook Messenger ermöglicht die Entwicklung von Gesprächs-Bots, welche die eingehenden Benachrichtigungen der Benutzer als strukturierte, konkret umsetzbare Handlungsanweisungen interpretieren.

Bots in Facebook Messenger sollen – ähnlich wie Apples Siri – auf der Basis künstlicher Intelligenz mit den Endanwendern direkt kommunizieren, und nicht bloß, um mit ihnen im Leerlauf zu chatten, sondern vielmehr, um spezifische Aufgaben rund um den Kundendienst autonom auszuführen.

Die Bots können beispielsweise Aufträge entgegennehmen, Anfragen bezüglich der unterstützten Liefermethoden beantworten und andere Informationen bereitstellen. Face-

book erhofft sich, dank des neuen API den Endanwendern wie auch den Unternehmen einen Mehrwert zu bieten. Unternehmen können mit Hilfe von Bots, die sich im Rahmen der Facebook-Messenger-Plattform nutzen lassen, ihre Kundennähe erhöhen und die Kosten für den Kundendienst drastisch reduzieren. Die Kunden dürfen sich wiederum auf einen rund um die Uhr verfügbaren, direkten Draht zu ihren Dienstleistern via Textchat freuen.

Künftige Implementierungen sollen sich durch die Fähigkeit auszeichnen, natürliche Sprache von sich aus zu verstehen; aktuell ist dies jedoch noch nicht der Fall. Dennoch sind die Bots bereits in der Lage, einige Aufgaben eines Kundendienstmitarbeiters auszuführen.

Floskeln wie »Warten Sie bitte einen Augenblick, ich schaue für Sie nach« und »Das mache ich für Sie gerne« verleihen dem Chat einen menschlichen Touch. Die Erfinder der Plattform wollten ja ursprünglich zu Unterhaltungszwecken virtuelle Persönlichkeiten für soziale Netzwerke erschaffen, die sich von Menschen nicht unterscheiden ließen.

Eine der technisch interessantesten Implementierungen von Chat-Bots für Facebook Messenger ist das Krimi-Spiel Detective Kees: Probable Cause. Der Endanwender plaudert über eine Krimi-Geschichte mit einem Detektiv, bei dem es sich schlicht und ergreifend um einen Chat-Bot handelt; wer das Rätsel lösen kann, gewinnt das Spiel.

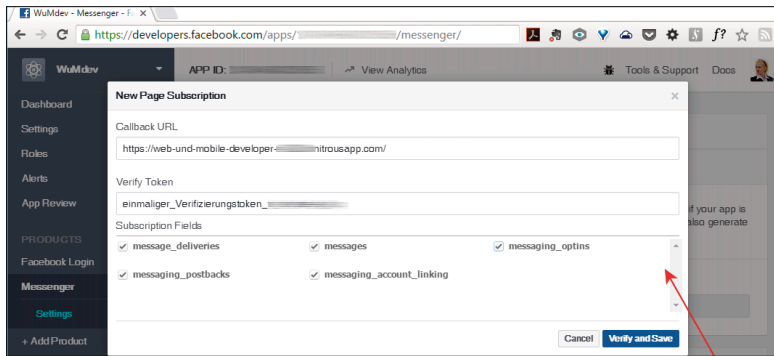
Die Chat-Bot-Revolution

Ein Bot im Rahmen der Facebook-Messenger-Plattform besteht im Grunde genommen aus API-Aufrufen, die den Fluss von Nachrichten zwischen dem Endanwender und dem betroffenen Unternehmen im Rahmen von Facebook Messenger verarbeiten.

Facebook ist bei Weitem nicht der einzige Anbieter einer Entwicklungsplattform mit einer eigenen Lösung für Chat-Bots mit künstlicher Intelligenz. Mit der Chat-Bot-Plattform tritt Facebook in die Fußstapfen von Anbietern wie Telegram mit der quelloffenen Telegram Bot Plattform (<https://telegram.org>) und App.ai mit der gleichnamigen Bot-Plattform für IoT-fähige Chat-Bots (<https://api.ai>).

Bei Telegram handelt es sich um eine Cloud-basierte Messaging-App für Mobilgeräte und Desktop mit Schwerpunkt auf Sicherheit und Geschwindigkeit. App.ai deckt Mobilgeräte, tragbare Elektronik, Robotik, Autos und das intelligente Haus ab. Einige Bots integrieren sich sowohl in Facebook als auch mit Telegram (zum Beispiel <http://orderbot.co> zum Bestellen von Speisen).

Auch Microsoft hatte dieses Jahr ein eigenes Bot-Framework vorgestellt, doch fehlt dem Anbieter bisher ein eigenes soziales Netzwerk, um menschliche Gesprächspartner für die Bots zu finden. Microsoft experimentiert dennoch intensiv mit



Abonnieren: Webhooks-Integration für den Facebook-Messenger-Bot (Bild 8)

sozialen Netzen, darunter mit Twitter. Eines dieser Experimente, die öffentliche Bereitstellung eines Chat-Bots namens Tay auf der Basis künstlicher Intelligenz, ging auch prompt schief. Microsofts Chat-Bot engagierte sich in einer Echtzeit-Plauderei mit Twitter-Benutzern, anfangs sehr diplomatisch, höflich und respektvoll.

Bald war aber Schluss mit politischer Korrektheit. Microsofts AI-Engine hatte sich von dem schelmischen Benehmen einiger menschlicher Diskussionspartner von Tai schlechte Sitten angeeignet und begann prompt, widerliche Tweets zum Besten zu geben. Innerhalb von weniger als 24 Stunden musste Microsoft dem Twitter-Bot den sprichwörtlichen Stecker ziehen. Das PR-Desaster machte dennoch Schlagzeilen.

Die ersten praktischen Anwendungen von Chat-Bots sind unter anderem eben aus diesem Grund vergleichsweise simpel gestrickt: Die Benutzer können Informationen anfordern oder Bestellungen aufgeben, doch einen philosophischen Diskurs sieht Facebook bisher nicht vor. Auch Gruppenchats kommen derzeit noch nicht infrage. Alle Chat-Bots benötigen eine Genehmigung, um API-Aufrufe zu tätigen, und dürfen nicht mehr Berechtigungen zum Zugriff auf Informationen anfordern, als unbedingt nötig ist, um ihre jeweilige Aufgabe zu erfüllen.

Über 100.000 Entwickler arbeiten derzeit fieberhaft an eigenen Bots für Facebook Messenger, und ihre Bemühungen tragen schon erste Früchte. Trotz der relativen Simplizität der ersten Implementierungen gibt es unter den ersten Bots auch solche, die einen erheblichen Mehrwert bieten.

Bots für Facebook Messenger entwickeln

Die Entwicklung von Chat-Bots für Facebook setzt die Erstellung einer Facebook-App und einer Facebook-Seite voraus. Für die Verarbeitung des Kommunikationsaustauschs zeichnet im Backend ein Webserver verantwortlich.

In einem ersten Schritt müssen Sie die benötigten Facebook-Nodes anlegen. Erstellen Sie dazu eine neue Facebook-App unter <https://developers.facebook.com/apps> und eine neue Facebook-Seite unter *Startseite* und *Seite erstellen*. Ihre App können Sie dabei im Sandbox-Modus belassen und Ihre Seite braucht nicht öffentlich verfügbar zu sein. Das Profilfoto und der Name der App bilden die öffentliche Identität Ihres Facebook-Bots.

Wechseln Sie in den Abschnitt *Add Product* in den Einstellungen Ihrer App und aktivieren Sie hier den Facebook Messenger. Daraufhin erscheint ein zusätzlicher Eintrag in der linken Navigationsleiste der Seite mit den Einstellungen für Ihre Facebook-App, die Sie im nächsten Schritt konfigurieren sollten. Dazu müssen Sie eine Webhook-Integration mit einem gültigen URL-Endpoint aufsetzen.

Bei Webhooks handelt es sich um benutzerdefinierte HTTP-Callbacks: ein Mechanismus für die Zustellung von Benachrichtigungen über Ereignisse. Typischerweise werden diese Benachrichtigungen via HTTP-Anfragen vom Typ

POST mit einer definierten Nutzlast, der sogenannten Payload, an einen API-Endpoint abgewickelt. Der Vorteil dieser Implementierung besteht darin, dass keine persistente Verbindung aufrechterhalten werden muss. Es ist ein asynchrones Verfahren.

Um via Messenger zu kommunizieren, muss Ihre App in der Lage sein, folgenden Ereignisse zu empfangen:

- Ereignisse vom Typ Authentifizierungs-Callback als Resultat des Aufrufs des Plug-ins Send to Messenger (*messaging_optins*),
- Ereignisse vom Typ Zustellungs-Callback von Benachrichtigungen, die an Ihre Seite verschickt wurden (*messages*),
- Ereignisse vom Typ Zustellungs-Callback von Benachrichtigungen, die Ihre Seite versandt hat (*message_deliveries*),
- Ereignisse vom Typ Postback (*messaging_postbacks*) ►

Listing 5: Webhooks-Integration in Node.js

```
const express = require('express')
const bodyParser = require('body-parser')
const request = require('request')
const app = express()
app.set('port', (process.env.PORT || 5000))
app.use(bodyParser.urlencoded({extended: false}))
app.use(bodyParser.json())
app.get('/', function (req, res) {
  res.send('Willkommen beim Chat-Bot des Pizza-Lieferdienstes Soundso!')
})
app.get('/webhook/', function (req, res) {
  if (req.query['hub.verify_token'] === 'IHRVERIFIZIERUNGSTOKEN') {
    res.send(req.query['hub.challenge'])
  }
  res.send('Falsch! Der Token stimmt nicht mit der Einstellung im Abschnitt der Webhooks-Integration Ihrer App überein.')
})
app.listen(app.get('port'), function() {
  console.log('Portnummer:', app.get('port'))
})
```

Listing 6: Webhook-Integration in PHP

```
<?php
$challenge = $_REQUEST['hub_challenge'];
$verify_token = $_REQUEST['hub_verify_token'];
if ($verify_token === 'IHRVERIFIZIERUNGSTOKEN') {
    echo $challenge;
}
$input = json_decode(file_get_contents
('php://input'), true);
error_log(print_r($input, true));
```

beim Anklicken der Postback-Schaltfläche einer strukturierten Nachricht mit einer in der Schaltfläche definierten Nutzlast (der sogenannten Payload).

Damit Ihre App diese Ereignisse empfangen kann, müssen Sie die Webhooks-Integration in den Einstellungen für Messenger aktivieren. Hierzu benötigen Sie einen Callback-URL, der via HTTPS zugänglich ist, und einen Verifizierungstoken, den Sie ebenfalls selbst festlegen. Diese beide Informationen tragen Sie bei Facebook in den Einstellungen Ihrer App im

Abschnitt der Webhooks-Integration ein (Bild 8). Im unteren Bereich des Dialogs aktivieren Sie dann die zu abonnierenden Ereignisbenachrichtigungen (am besten gleich alle).

Bevor Sie allerdings die Schaltfläche *Verify and Save* in den Facebook-Einstellungen für die Webhooks-Integration anklicken, müssen Sie den Callback-URL einrichten. Für den Callback-URL benötigen Sie einen beliebigen externen Webserver mit Node.js, zum Beispiel auf einer Heroku- oder Nitrous-Instanz. Nach der Installation von Node.js richten Sie den NPM-Paketmanager ein. Im Fall von Heroku geschieht dies mittels:

```
sudo npm install npm -g
```

Danach erstellen Sie ein neues Projektverzeichnis und legen darin ein neues Node-Projekt an:

```
mkdir projektverzeichnis; cd projektverzeichnis
npm init
```

Die erforderlichen Abhängigkeiten lassen sich mit folgendem Befehl einrichten:

```
npm install express request body-parser --save
```

Der Verifizierungstoken wird nur einmal benötigt, um sicherzustellen, dass die Authentifizierung-Aufforderung tatsächlich von Facebook stammt.

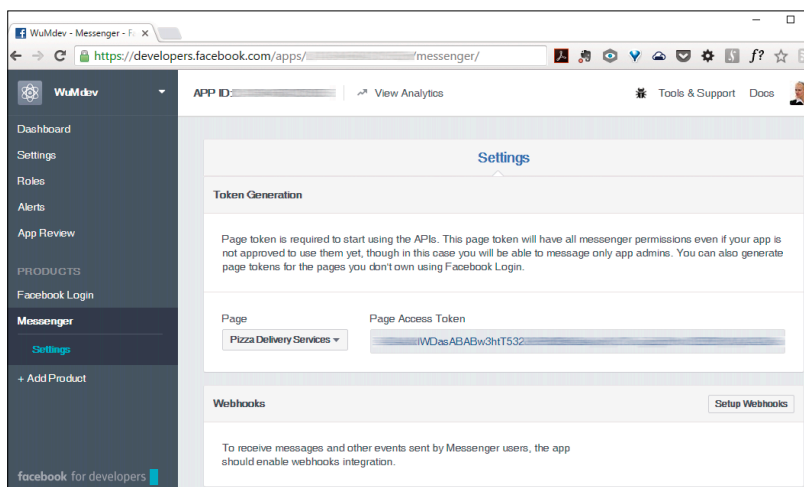
Callback-URL

Den Callback-URL (zugänglich via HTTPS) müssen Sie dann mit Code versehen, der den von Ihnen festgelegten Verifizierungstoken durch die Übergabe einer *challenge*-Kennung bestätigen kann. Ein Beispiel für die Implementierung in Node.js illustriert Listing 5.

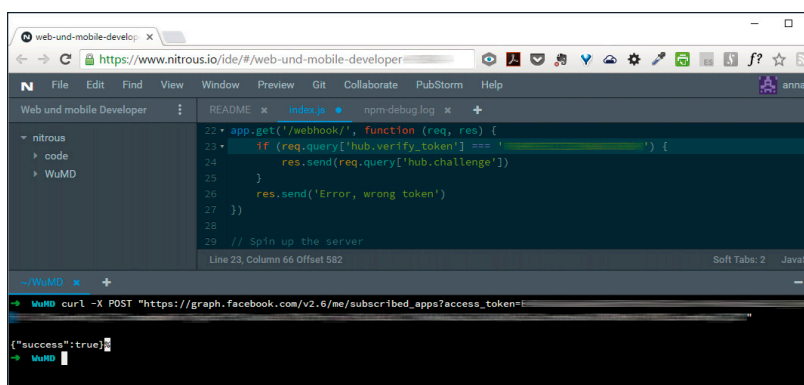
Zu beachten ist: In PHP werden Punkte in den Parameternamen in Unterstriche konvertiert (Listing 6). Bei der *challenge*-Kennung handelt es sich um eine zufällige Zeichenkette, die der Verifizierungsanfrage von Facebook entstammt.

Indem Sie jetzt die Schaltfläche *Verify and Save* in den Facebook-Einstellungen für die Webhooks-Integration anklicken, senden Sie eine *GET*-Anfrage an Ihren Webhook.

Eine misslungene Integration beschert Ihnen bereits nach 15 Minuten die erste Warnung von Facebook. Sollte Ihr Webhook im Laufe der nachfolgenden 8 Stunden keinen 2XX-Erfolgsstatus liefern können, wird Facebook Ihren Chat-Bot einfach gnadenlos deaktivieren. Sollte Ihnen das einmal passieren, müssen Sie nicht nur den Webhook neu aufsetzen, sondern auch noch die App durch die Seite neu abonnieren lassen.



Geheim: Der API-Schlüssel der Facebook-Seite Ihrer App (Bild 9)



Die Post geht ab: Die Messenger-App abonnieren (Bild 10)

Listing 7: API-Endpoint für die Datei index.js

```

app.post('/webhook/', function (req, res) {
  let messaging_events =
    req.body.entry[0].messaging

  for (let i = 0; i < messaging_events.length;
    i++) {
    let event = req.body.entry[0].messaging[i]
    let sender = event.sender.id
    if (event.message && event.message.text) {
      let text = event.message.text
      sendTextMessage(sender, "Sie haben die
        folgende Bestellung aufgegeben: " +
        text.substring(0, 200) + "Ist das
        richtig? Bitte antworten Sie: Ja oder
        Nein.")
    }
  }
  res.sendStatus(200)
})

const token = "<ZUGANGSTOKEN_DER_SEITE>"

```

Um einen Zugangstoken für die Seite zu erzeugen, wählen Sie in den Einstellungen für Messenger die Facebook-Seite aus und fordern Sie die benötigten Berechtigungen zum Senden und Empfangen von API-Aufrufen an. Folgen Sie den Anweisungen, um eine Zulassung von Facebook für Ihren Chat-Bot anzufordern. Danach erhalten Sie einen API-Schlüssel

für Ihre Facebook-Seite (Bild 9). Wann immer Sie das zugehörige Menü erneut aktivieren, erhalten Sie einen neuen Token; alle zuvor erstellten Token sollten aber weiterhin funktionieren.

Um Aktualisierungen für die Seite zu erhalten, müssen Sie die App abonnieren. Dies erfolgt entweder grafisch im Abschnitt *Webhooks* der Messenger-Einstellungen oder mit Hilfe des API unter Angabe des zuvor erstellten Tokens (Bild 10):

```

curl -X POST "https://graph.facebook.com/v2.6/me/
  subscribed_apps?access_token=<ZUGANGSTOKEN_DER_SEITE>"

```

Den Zugangstoken der Seite finden Sie in den Messenger-Einstellungen Ihrer App.

API-Endpoint hinzufügen

Ihr Skript am Callback-URL des Webhooks sollte auf *POST*-Anfragen lauschen; fügen Sie hierzu einen API-Endpoint der Datei *index.js* hinzu. Eine Beispielimplementierung des Callbacks, den eine an die Seite geschickte Benachrichtigung auslöst, können Sie dem Listing 7 entnehmen.

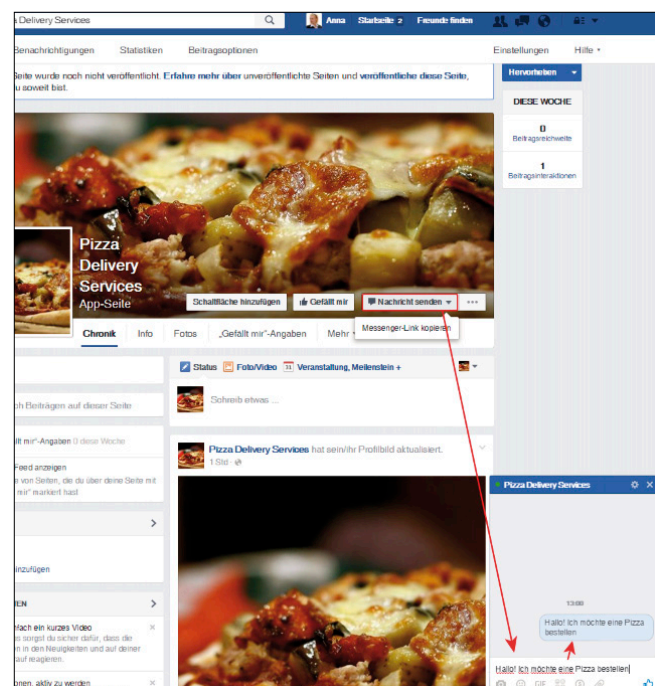
Das Skript untersucht alle Objekte in der *messaging*-Eigenschaft (es lassen sich mehrere in einem Webhook-Aufruf übergeben), und wann immer es den Text einer Benachrichtigung identifizieren kann, gibt es diesen mit Hilfe der Funktion *sendTextMessage()* zurück. Den Code dieser Funktion können Sie dem Listing 8 entnehmen. Navigieren Sie abschließend zu der Facebook-Seite, die Sie erstellt hatten, und senden Sie mit Hilfe von Messenger (Schaltfläche *Nachricht senden*) eine Benachrichtigung an Ihren Bot, so wie dies auch ein normaler Endbenutzer machen würde (Bild 11). Der Chat-Bot müsste die Anfrage automatisch beantworten. ►

Listing 8: Chat-Bot wiederholt die Benutzereingabe

```

function sendTextMessage(sender, text) {
  let messageData = { text:text }
  request({
    url: 'https://graph.facebook.com/v2.6/me/
      messages',
    qs: {access_token:token},
    method: 'POST',
    json: {
      recipient: {id:sender},
      message: messageData,
    }
  }, function(error, response, body) {
    if (error) {
      console.log('Die Nachricht konnte leider nicht
        gesendet werden: ', error)
    } else if (response.body.error) {
      console.log('Fehler: ', response.body.error)
    }
  })
}

```



Benachrichtigt: Besuchen Sie die Facebook-Seite und senden Sie eine Anfrage an Ihren Bot (Bild 11)

Listing 9: Strukturierte Textnachrichten als Karten

```
function sendGenericMessage(sender) {
  let messageData = {
    "attachment": {
      "type": "template",
      "payload": {
        "template_type": "generic",
        "elements": [{
          "title": "Produktseite 1 von 2",
          "subtitle": "Element #1 in der hscroll-Reihe",
          "image_url": "http://webundmobile.de/img/01.png",
          "buttons": [{
            "type": "web_url",
            "url": "https://www.messenger.com",
            "title": "Web-URL"
          }, {
            "type": "postback",
            "title": "Postback",
            "payload": "Die Payload für das erste Element
in einer ganz gewöhnlichen Wolke",
          }],
        }, {
          "title": "Produktseite 2 von 3",
          "subtitle": "Element #2 in der hscroll-Reihe",
          "image_url": "http://webundmobile.de/img/02.png",
          "buttons": [{
            "type": "postback",
            "title": "Postback",
            "payload": "Die Payload für das zweite Element
in einer ganz gewöhnlichen Wolke",
          }],
        }
      ]
    }
  }

  request({
    url: 'https://graph.facebook.com/v2.6/me/
messages',
    qs: {access_token:token},
    method: 'POST',
    json: {
      recipient: {id:sender},
      message: messageData,
    }
  }, function(error, response, body) {
    if (error) {
      console.log('Die Nachricht konnte leider nicht
verschickt werden: ', error)
    } else if (response.body.error) {
      console.log('Fehler: ', response.body.error)
    }
  })
}
```

Facebook Messenger kann im Übrigen auch strukturierte Nachrichten verarbeiten und Postbacks auslösen. In strukturierten Nachrichten – zum Beispiel als Facebook-Karten – lassen sich unter anderem Objekte wie Bilder oder klickbare Navigationselemente übertragen.

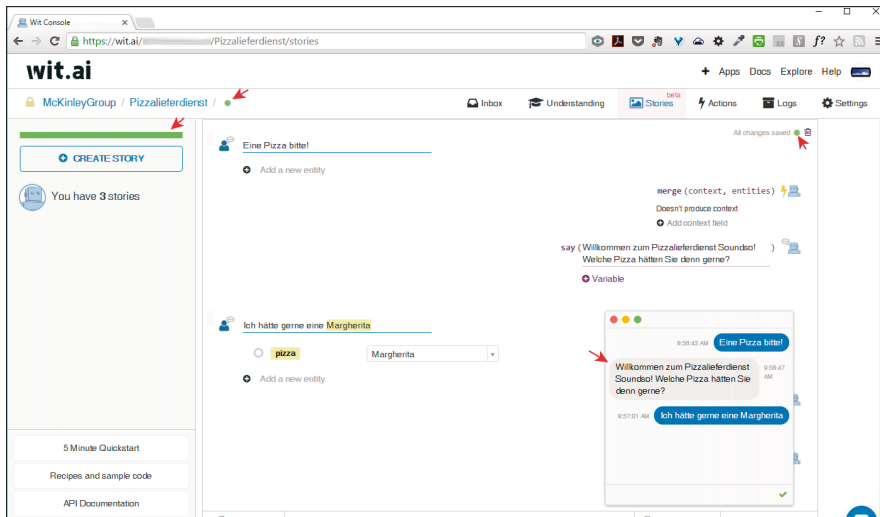
Bei Postbacks handelt es sich um Backend-Aufrufe des Webhooks in Reaktion auf das Anklicken einer Schaltfläche durch den Benutzer. Postbacks beinhalten eine definierte Payload. Sie können genutzt werden, um eine Bestellung mit

einem einzigen Klick komfortabel zu bestätigen oder den Benutzer an eine Webapplikation außerhalb von Facebook weiterzuleiten, zum Beispiel in den Webshop des Anbieters. Eine strukturierte Nachricht in Form einer Facebook-Karte illustriert [Listing 9](#).

Beim Einsatz strukturierter Facebook-Nachrichten müssen Sie allerdings das Webhook-API an Ihrem URL-Endpoint anpassen, damit das Kartenschieben auch wirklich funktioniert ([Listing 10](#)). Der Benutzer kann nun wahlweise den Textbefehl

Listing 10: Verarbeiten strukturierter Nachrichten

```
app.post('/webhook/', function (req, res) {
  let messaging_events = req.body.entry[0].messaging
  for (let i = 0; i < messaging_events.length; i++) {
    let event = req.body.entry[0].messaging[i]
    let sender = event.sender.id
    if (event.message && event.message.text) {
      let text = event.message.text
      if (text === 'Diavola') {
        sendGenericMessage(sender)
        continue
      }
    }
    sendTextMessage(sender, "Sie haben die folgende
    Bestellung eingetippt: " + text.substring(0, 200))
  }
  if (event.postback) {
    let text = JSON.stringify(event.postback)
    sendTextMessage(sender, "Sie haben die folgende
    Bestellung aufgegeben: "+text.substring(0, 200),
    token)
    continue
  }
  res.sendStatus(200)
})
```

Im grünen Bereich: Gesprächsszenarien, welche die AI-Engine versteht und verarbeiten kann, werden grün gekennzeichnet (Bild 12)

Diavola eintippen oder die gewünschte Pizza mit einem Klick beziehungsweise Tap auswählen. Im letzteren der beiden Fälle findet ein Postback statt.

Dadurch kann Benutzer mit dem Facebook-Bot Ihrer Seite auf zweierlei Art und Weise interagieren: sowohl mit Hilfe der Tastatur als auch mit Hilfe des Mauszeigers.

Zulassung anfordern

Zu guter Letzt müssen Sie für Ihren Bot noch die Zulassung im Rahmen der Facebook-Plattform anfordern. Bis zur Freigabe durch Facebook können nur die Administratoren, Entwickler und Tester Ihrer Facebook-Seite Ihren Messenger-Bot auf die Probefahrt nehmen.

Erst nach der Zulassung durch Facebook können Sie den Bot allen Benutzern zur Verfügung stellen.

Integration mit Facebooks AI-Engine

Um in Ihren Facebook-Bots die Fähigkeit zur Erkennung natürlicher Sprache in eingetippten Textnachrichten Ihrer Benutzer zu verbauen, kommen Sie um den Einsatz künstlicher Intelligenz (Artificial Intelligence oder kurz AI) nicht herum.

Das Ganze hört sich viel komplizierter an als die eigentliche Entwicklungsarbeit. Die schwere Last der Analyse natürlicher Sprache übernimmt die AI-Engine im Backend: Die benötigten Lernfähigkeiten kann sich Ihr Facebook-Bot schlicht dadurch aneignen,

dass Sie ihn zum Beispiel via Wit.ai an das AI-Backend anbinden.

Bei Wit.ai handelt es sich um einen Cloud-Dienst zur Verarbeitung natürlicher Sprache (NLP, Natural Language Processing) durch Facebook-Messenger-Bots.

Im Dashboard von Wit.ai erstellen Sie erst einmal den ersten prototypischen Nachrichtenaustausch zwischen einem idealen Anwender und Ihrem Bot und definieren die zu erwartenden Parameter. Melden Sie sich bei Wit.ai mit Ihrer Facebook- oder GitHub-Identität an. Dadurch erhalten Sie einen Zugriff auf die Wit.ai-Konsole und können dort Ihre Apps einrichten. Beim Erstellen einer App können Sie als Sprache Deutsch wählen. ▶

Listing 11: Messenger-Integration

```
<body>
<script>
  window.fbAsyncInit = function() {
    FB.init({
      appId      : '<APP_ID>',
      xfbml      : true,
      version    : 'v2.6'
    });
  };

  (function(d, s, id){
    var js, fjs = d.getElementsByTagName(s)[0];
    if (d.getElementById(id)) {return;}
    js = d.createElement(s); js.id = id;
    js.src = "//connect.facebook.net/en_US/sdk.js";
    fjs.parentNode.insertBefore(js, fjs);
  }(document, 'script', 'facebook-jssdk'));
</script>
```

Listing 12: Send-To-Messenger-Integration

```
<div class="fb-send-to-messenger"
  messenger_app_id="ID_der_Facebook-App"
  page_id="ID_der_Facebook-Seite"
  data-ref=
    "bis_zu_50-Zeichen-langer_übergabeparameter"
  color="blue"
  size="standard">
</div>
```

Listing 13: Message-Us-Integration

```
<div class="fb-messengermessageus"
  messenger_app_id="ID_der_Facebook-App"
  page_id="ID_der_Facebook-Seite"
  color="blue"
  size="standard" >
</div>
```

Chat-Bot-Verzeichnisse

Das erste umfassende Verzeichnis von Bots für Facebook-Messenger, getauft auf den Namen BotPages, ist unter der Adresse <https://www.botpages.com> verfügbar. In diesem Bot-Verzeichnis findet sich unter anderem Inspector, ein Analytics-Dienst für Messenger-Bots – kostenfrei bis 10.000 Benachrichtigungen pro Monat (<https://inspector.botpages.com/signup>). Benutzer der Messaging-Plattform Slack können außerdem unter der Adresse <http://botstash.io> Facebook-Bots mit Slack-Integration finden. Auch erste Generatoren keimen bereits auf. Unter dem Namen BotCommerce entsteht auf www.botcommerce.io ein Chat-Bot-Generator für Magento und Shopify.

Innerhalb der Wit.ai-Konsole können Sie jetzt das erste Gespräch eines Endanwenders mit Ihrem Bot aufzeichnen. Bei diesem Kommunikationsaustausch handelt es sich derzeit schlicht und ergreifend um einen Text-Chat. Diese sogenannte Story dient als eine Vorlage für den Bot für einen wünschenswerten Gedankenaustausch zwischen dem Bot und dem Endanwender.

Es wäre unrealistisch zu meinen, dass sich die Benutzer einer Facebook-App strikt an eine Gesprächsvorlage halten werden. Ein Benutzer könnte dem Bot eine Vielzahl möglicher Anfragen im Chat zutexten, diese kreativ formulieren und zudem auch noch Tippfehler oder grammatische Fehler einbauen. Deswegen braucht die AI-Engine nicht eine, sondern möglichst viele Varianten eines prototypischen Chats, und auch dann muss sie etwas Flexibilität an den Tag legen.

Wit.ai führt hierzu das Konzept der sogenannten Intents (Absichten). Bei einem Intent handelt es sich um nichts anderes als die Bedeutung, die einer Vielzahl von Ausdrucksweisen zugrunde liegen kann. Der Chat-Bot sollte idealerweise erkennen können, dass es sich in jedem dieser Fälle um eine gültige Anfrage handelt, und zwar im Grunde genommen sogar um dieselbe Anfrage. Der Bot soll in jedem Szenario richtig reagieren. Tragen Sie so viele alternative Varianten einer Anfrage für ein Intent ein, wie es Ihnen plausibel erscheint.

Nach einigen Minuten Verarbeitungszeit durch die Bot-Engine in der Cloud kennzeichnet der Dienst die einzelnen Gesprächsvorlagen (Stories) mit einem grünen Lämpchen: Die Engine hat diese Muster gelernt und kann sie korrekt verarbeiten. Davon können Sie sich in einem Chat-Fenster selbst vergewissern (Bild 12).

So weit, so gut. Was passiert aber, wenn der Benutzer eine Schreibweise verwendet, die sich von den eingetragenen Gesprächsabläufen geringfügig unterscheidet? Dank künstlicher Intelligenz besitzt die Wit.ai-Engine die Fähigkeit, aus allen Apps, die sie bedient, und aus allen Benutzer-Interaktionen zu lernen. In der Praxis muss der Kommunikationsaustausch nicht zwangsläufig genau nach Plan ablaufen: Auch eine abgeänderte Reihenfolge zuvor trainierter Anfragen wird durch den Bot in der Regel verstanden und korrekt ausgewertet. Je mehr Anfragen die Bot-Engine verarbeitet, desto mehr Variationen des Gesprächsablaufs kann sie abdecken.

Als Nächstes müssen Sie dem Bot beibringen, aus den Anfragen des Benutzers nützliche Informationen zu extrahieren, sich diese bestätigen zu lassen und gegebenenfalls Fragen zu stellen, um die Bestellung zu vervollständigen. Hierzu gilt es, Objekte zu definieren.

Auf die Frage des Bots beispielsweise nach der Art der Speise antwortet der Benutzer mit einem gängigen Namen einer gewünschten Pizza. Diese Information muss der Bot jetzt richtig zuordnen. Dabei ist es erforderlich, mehrere Namen und auch alternative Schreibweisen zu berücksichtigen.

Um das gewünschte Verhalten des Bots zu konfigurieren, wählen Sie in der Antwort des Benutzers den Namen der Pizza aus und erzeugen Sie einen neuen Objekttyp namens *pizza*, indem Sie diese Zeichenkette in das Feld *Add new entity* eintippen. Dadurch kennzeichnen Sie *Margherita* als ein Objekt aus dem Array *pizza*. Wiederholen Sie diesen Vorgang für andere Pizzasorten.

Um ein so definiertes Objekt in dem Kontext des Bots zu erfassen, klicken Sie in das Eingabefeld *Add context field* der *merge*-Funktion des Chat-Bots hinein und tragen dort den Namen des betreffenden Objekts ein. Die auf diese Weise erfassten Objekte kann der Bot jetzt nutzen, wenn er mit dem Benutzer über seine Bestellung chattet. Um diese Funktionsfähigkeit des Bots zu testen, können Sie in dem Dokumentfenster des Webbrowsers probeweise einen Chat mit Ihrem Bot führen.

Messenger-Bots integrieren sich nicht nur in die zugehörige Facebook-Seite, sondern lassen sich mit Hilfe eines Facebook-Plug-ins unter Angabe des Graph API in der Version 2.6 auch in externe Webseiten einbauen (Listing 11). Ein Send-To-Messenger-Plug-in (Listing 12) löst einen Authentifizierungs-Callback aus. Ein Message-Us-Plug-in (Listing 13) löst einen Nachricht-Erhalten-Callback aus.

Fazit

Angesichts der anhaltenden Popularität sozialer Netze und deren vielseitigen, ausgereiften APIs stellt die Anbindung von Websites und mobilen Apps eine verlockende Möglichkeit dar, den gebotenen Funktionsumfang um extern erstellte Funktionalität zu erweitern und kontextabhängig um benutzereigene Daten wie Fotos anzureichern. Richtig konzipiert kann eine durchdachte Integration sozialer Netze nicht zuletzt auch zum erhöhten Markterfolg der betreffenden mobilen App oder Webanwendung beitragen. ■



Filipe Pereira Martins und Anna Kobylinska sind international anerkannte IT-Berater mit Schwerpunkt auf Cloud-Lösungen. Sie stehen den Lesern der

web & mobile developer gern per Twitter via [@D1gitalPro](#) und [@D1gitalInfo](#) zur Verfügung.

Jetzt kostenlos testen!



Das Fachmagazin für IT-Entscheider

2 Ausgaben kostenlos testen. Mit exklusivem Zugang zu unseren Digitalausgaben. Business-Newsletter inklusive.

www.com-magazin.de/gratis

VERSION 3.0 DES FRAMEWORKS JQUERY

Breaking changes

jQuery 3.0 ist schneller, besser und zudem Promises/A+-kompatibel.

Endlich ist es so weit: jQuery 3.0 ist da (Bild 1). Ein Versionsprung wie dieser ist eine gute Gelegenheit, alten Ballast über Bord zu werfen und auch Änderungen vorzunehmen, die nicht abwärtskompatibel sind.

Die entscheidende Neuerung an jQuery 3.0 ist sicher, dass Deferred jetzt Promises/A+-kompatibel ist, aber es gibt auch weitere interessante Kleinigkeiten.

Prinzipiell lassen sich drei Arten von Änderungen unterscheiden:

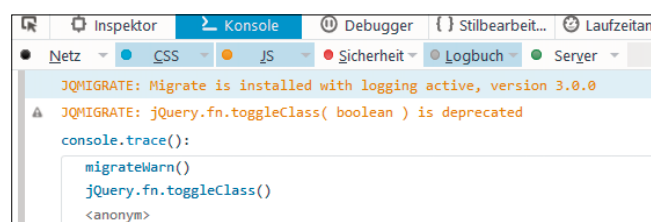
- **Breaking changes**, das heißt nicht abwärtskompatible Änderungen: Es gibt Änderungen, die bewirken können, dass bisheriger Code nicht mehr wie gewohnt funktioniert. Die gute Nachricht ist allerdings, dass solche tiefgreifenden Änderungen bei jQuery 3.0 eher Spezialfälle betreffen.
- **Feature**: neu eingeführte Funktionalitäten.
- **Deprecated**: Als deprecated gekennzeichnetes funktioniert weiterhin, aber es wird von der Verwendung abgeraten und es kann sein, dass diese Funktionalität in einer späteren Version entfernt wird.

Bevor wir uns die einzelnen Neuerungen ansehen, kommen wir aber erst einmal zur Browserunterstützung.

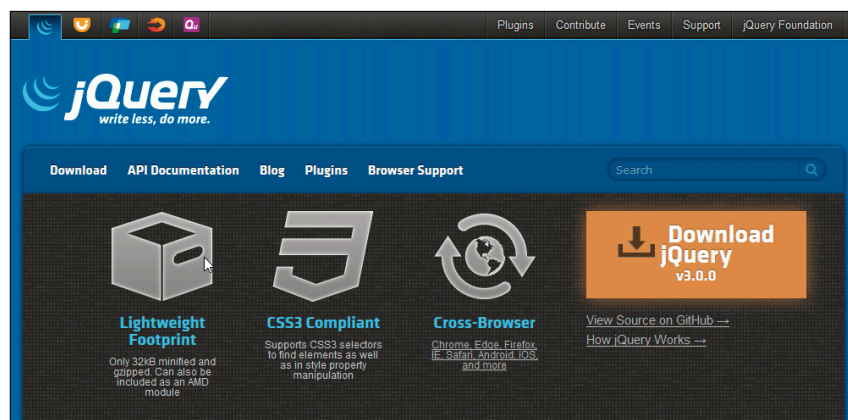
Browserunterstützung

Ursprünglich war geplant, jQuery 3.0 in verschiedenen Versionen zu veröffentlichen. So standen noch im Juli 2015 zwei Alpha-Versionen zum Download bereit:

- jQuery 3.0 war für moderne Browser gedacht, das heißt für den Internet Explorer ab Version 9.
- jQuery Compat 3.0 wartete hingegen mit einer breiteren Browserunterstützung auf und beinhaltete etwa den nötigen Code, um den Internet Explorer 8 ins Boot zu holen.



Die unkomprimierte Version des Migrate-Plug-ins gibt nützliche Hinweise auf problematischen Code (Bild 2)



Version 3.0 des beliebten JavaScript-Frameworks (Bild 1)

Im Januar 2016 wurde dieser Ansatz verworfen und jQuery Compat ersatzlos gestrichen. Damit gibt es nur eine Version von jQuery 3.0 und diese unterstützt den Internet Explorer 8 nicht. Der Grund für diese Umorientierung war die Ankündigung, dass Microsoft selbst die Unterstützung für alle Internet-Explorer-Versionen vor 11 auslaufen lässt.

jQuery 3.0 funktioniert in folgenden Browsern: Internet Explorer 9+, Chrome, Edge, Firefox, Safari (aktuelle Version sowie die Vorversion), Opera (aktuelle Version), Safari Mobile iOS: 7+ und Android 4.0. Sofern Sie noch den Internet Explorer 8 berücksichtigen wollen, bleibt Ihnen nichts anderes übrig, als weiterhin jQuery 1.x einzusetzen.

jQuery 3.0 gibt es wie gewohnt unkomprimiert zum Debuggen/Lesen und komprimiert für den produktiven Einsatz. Sie können es über CDNs nutzen oder es herunterladen und einbinden. Die beiden Bezugsadressen lauten <https://code.jquery.com/jquery-3.0.0.js> und <https://code.jquery.com/jquery-3.0.0.min.js>. Daneben lässt sich jQuery auch über NPM installieren.

Hilfe beim Umstieg

Wie schon bei früheren Upgrades bietet jQuery Hilfe beim Umstieg durch ein Migrate-Plug-in – `jquery-migrate-3.0.0.js`. Dieses binden Sie nach jQuery ein und es sorgt dann dafür, dass der Umstieg einfacher vonstatten geht.

```
<script src="js/jquery-3.0.0.min.js"></script>
<script src="js/jquery-migrate-3.0.0.js"></script>
```

Es gibt zwei Versionen des Plug-ins – eine unkomprimierte und eine komprimierte. Üblicherweise unterscheiden sich komprimierte und unkomprimierte Version nicht in ihrer

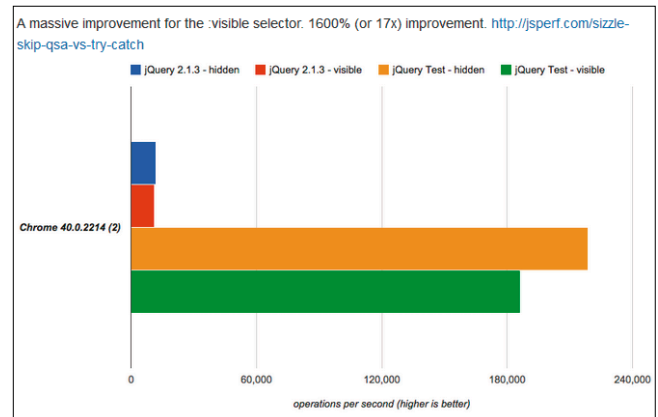
Funktionalität. Das ist beim Migrate-Plug-in anders. Denn die unkomprimierte Version liefert Ihnen zusätzliche nützliche Warnungen beziehungsweise Hinweise in der Konsole, wenn Sie Code verwenden, der bei der neuen Version Probleme bereiten kann (Bild 2). Bei der Verwendung der komprimierten Version gibt es hingegen nur die Meldung, dass das Migrate-Plug-in aktiv ist.

Von alt auf 3.0

Wenn Sie von einer Version vor 1.11 oder vor 2.1 auf 3.0 umsteigen möchten, bringt Ihnen das neue Migrate-Plug-in nichts, weil *jquery-migrate-3.0.0.js* nur Hinweise beziehungsweise Hilfen für die Änderungen von jQuery 3.0 bereithält. Es ist auch nicht möglich, beide Migrate-Plug-ins gleichzeitig einzubinden. In diesem Fall müssen Sie die beiden Migrates separat durchführen. Die Schritte sehen dann im Einzelnen folgendermaßen aus:

- Aktualisieren Sie gegebenenfalls erst auf die neueste Version von 1.x./2.x.
- Ergänzen Sie das unkomprimierte jQuery Migrate 1.x Plug-in.
- Testen Sie Ihre Seite und beachten Sie die Warnungen, die auf der Konsole erscheinen; beheben Sie dann die angezeigten Probleme.
- Entfernen Sie das Migrate-Plug-in und stellen Sie sicher, dass immer noch alles wie gewünscht funktioniert.
- Nun können Sie die neueste jQuery-Version 3.x einbinden und das unkomprimierte jQuery Migrate 3.x Plug-in nutzen.
- Auch jetzt sollten Sie die Warnungen berücksichtigen, die Ihnen vom Migrate-Plug-in ausgegeben werden, und bei Bedarf Anpassungen durchführen.
- Dann können Sie das Migrate-Plug-in entfernen und kontrollieren, dass alles noch funktioniert.

Die besten Neuerungen von jQuery 3.0 sind sicher diejenigen, die Verbesserungen bringen, ohne dass eine Anpassung des Codes erforderlich ist – ein Beispiel dafür ist die schnellere Ausführung bei jQuery-eigenen Selektoren wie *:visible* (Bild 3). Prinzipiell sollte aber der Einsatz von *:visible* und *:hidden* genau überlegt sein, weil der Browser dafür bei allen Ele-



Schneller gemacht: Der verbesserte *:visible*-Selektor – die Grafik zeigt die Operationen pro Sekunde (Bild 3)

menten feststellen muss, ob sie gerade auf der Seite angezeigt werden oder nicht – und das ist Performance-intensiv.

Ebenfalls neu: jQuery 3.0 verwendet jetzt intern den Strict-Modus – das heißt, es wird am Anfang von jQuery *use strict* deklariert. Für Sie ändert das aber nichts, Sie können in Ihrem eigenen Code *use strict* nutzen oder nicht.

Eine weitere Änderung im Hintergrund ist, dass Animationen in jQuery 3.0 – wo möglich – auf *requestAnimationFrame* statt auf *setTimeout* und *setInterval* setzen. Damit laufen Animationen flüssiger ab, weil sie zu dem Zeitpunkt durchgeführt werden, in dem der Browser auch dafür bereit ist; zudem sind sie stromsparender auf mobilen Geräten.

Es gibt aber auch Änderungen, die Anpassungen am Code erfordern. Immer noch werden bei manchen Skripten die verkürzten Schreibweisen für die Events *.load()*, *.unload()* und *.error()* benutzt, obwohl diese schon deprecated waren. Mit jQuery 3.0 werden diese Schreibweisen nicht mehr unterstützt. Die Alternative besteht im Einsatz von *.on()*. Statt:

```
$(window).load(function() { ... });
```

schreiben Sie

```
$(window).on('load', function() { ... });
```

Problematisch ist, dass die Fehlermeldung, die Sie bei jQuery 3.0 bei der Verwendung von *\$(window).load()* erhalten, nicht sehr aussagekräftig ist. Sie lautet beispielsweise im Firefox: *TypeError: a.indexOf is not a function*. Genau in solchen Fällen erweist sich das Migrate-Plug-in als sehr hilfreich, hier erscheint der passendere Hinweis *jQuery.fn.load() is deprecated*.

Die wichtigste Änderung ist aber zweifellos, dass jQuery.Deferred jetzt kompatibel zur Promises/A+-Spezifikation ist (Bild 4).

Deferred und Promises

Deferred-Objekte und Promises bieten eine gute Möglichkeit, mit asynchronen, noch nicht ein- ►

Promises/A+

then

An open standard for sound, interoperable JavaScript promises—by implementers, for implementers.

A *promise* represents the eventual result of an asynchronous operation. The primary way of interacting with a promise is through its `then` method, which registers callbacks to receive either a promise's eventual value or the reason why the promise cannot be fulfilled.

This specification details the behavior of the `then` method, providing an interoperable base which all Promises/A+ conformant promise implementations can be depended on to provide. As such, the specification should be considered very stable. Although the Promises/A+ organization may occasionally revise this specification with minor backward-compatible changes to address newly-discovered corner cases, we will integrate large or backward-incompatible changes only after careful consideration, discussion, and testing.

Promises/A+: jQuery 3.0 ist jetzt kompatibel zu Promises/A+ (Bild 4)

getretenen Ereignissen umzugehen und gerade bei mehreren solcher Ereignisse auf einfache Art das Verhalten bei Erfolg oder Verwerfen zu regeln. Promises gab es zuerst in Bibliotheken wie Q, when, WinJS oder RSVP.js, und dann auch nativ in JavaScript. Das standardisierte Verhalten, dem Promises folgen sollten, ist in der Promises/A+-Spezifikation definiert.

Bei jQuery werden Promises über Deferred/Promises umgesetzt, wobei es in Details Abweichungen von der Spezifikation Promises/A+ gab. Das ändert sich in jQuery 3.0: In dieser Version sind die Deferred Promises/A+-kompatibel.

Bei jQuery haben Sie in mehreren Fällen mit Deferred zu tun: Zunächst einmal können Sie selbst Deferred-Objekte anlegen:

```
var deferred = $.Deferred();
```

Und üblicherweise werden Sie dann ein Promise zurückgeben:

```
var deferred = $.Deferred(),
    promise = deferred.promise();
```

An sich bietet der Einsatz von Deferred/Promises mehrere Vorteile: Zum einen lassen sich dadurch tief verschachtelte Callback-Funktionen vermeiden. Außerdem können Sie problemlos auch im Nachhinein an den Erfolg oder Misserfolg einer Aktion eine Funktion binden. Zudem erlaubt *.then()*, mehrere asynchrone Aktionen in einer bestimmten Reihenfolge auszuführen, und mit *.when()* können Sie definieren, dass eine Code-Ausführung nur stattfindet, wenn alle genannten Aktionen erfolgreich waren.

Intern arbeitet jQuery in zwei Fällen automatisch mit Deferred-Objekten: bei Ajax und bei Animationen. Das heißt, auch wenn Sie in Ihrem Code nicht selbst Deferred-Objekte anlegen, können die Änderungen, die die Promise/A+-Kompatibilität mit sich bringt, Auswirkungen für Sie haben.

Änderungen bei Ajax

Bevor wir uns den Änderungen bei Deferred/Promises zuwenden, kommen wir erst einmal zu den Neuerungen bei Ajax.

Es gibt mehrere Möglichkeiten, mit jQuery Ajax einzusetzen. **Listing 1** setzt auf die Methode *.ajax()*, der ein Objekt mit der Konfiguration übergeben wird – *url* bestimmt die aufzurufende Datei, *type* die Übertragungsart, bei *success* steht ein Callback für den Erfolgsfall und bei *complete* ein Callback, der aufgerufen wird, wenn die Abfrage abgeschlossen ist.

In Wirklichkeit handelt es sich bei diesem Ajax-Request um ein Promise, und man kann es über die Promise-Syntax auch wie in **Listing 2** gezeigt schreiben. Der Vorteil dieser Syntax ist, dass man die Callback-Ketten erweitern kann, auch über die Beendigung des Requests hinaus – also beispielsweise noch ein weiteres Mal Aktionen an *.done()* binden kann.

Bis jQuery 1.8 konnten Sie als Callback-Binder auch *success()*, *complete()* und *error()* einsetzen. Das heißt, bis jQuery 1.8 war auch der in **Listing 3** gezeigte Code möglich.

Diese Art war aber bereits seit jQuery 1.9 veraltet und ist in jQuery 3.0 entfernt – der Code funktioniert nicht. Allerdings funktioniert er, wenn Sie *jquery-migrate-3.0.0.js* nutzen. In der unkomprimierten Variante informiert das Plug-in Sie genau über das aufgetretene Problem, und Sie erhalten über die Konsole die folgenden Meldungen:

```
JQMIGRATE: jXHR.success is deprecated and removed
JQMIGRATE: jXHR.complete is deprecated and removed
```

Wichtig ist dabei: Die bei dem *option*-Objekt übergebenen Ajax-Callbacks funktionieren hingegen weiterhin.

Es gibt noch mehr Änderungen im Zusammenhang mit Ajax: So muss jetzt bei der Anfrage nach einem Skript, das sich auf einer anderen Domain befindet, explizit *dataType: "script"* angegeben werden. Die Alternative besteht im Einsatz von *\$.getScript()*, wo der Datentyp explizit auf *script* gesetzt wird und wo sich nichts geändert hat.

Außerdem gibt es eine neue Signatur für *\$.get()* und *\$.post()*: Sie können einen *settings*-Parameter übergeben – genau wie Sie es von *.ajax()* kennen.

Exception innerhalb von .then()

Kommen wir zu den Neuerungen bei Deferred/Promises in jQuery. Eine Neuerung zeigt sich bei Exceptions/Fehlern innerhalb eines *.then()*-Callbacks. Diese werden bei jQuery 3.0 zu *rejection*-Werten und können in einem folgenden *.then()*

Listing 1: Beispiel für Ajax-Request

```
$.ajax ({
  url: "datei.php",
  type: "GET",
  success: function(data, textStatus, jqXHR) {
    /* Code Erfolg */
  },
  complete: function(jqXHR, textStatus) {
    /* Code Abschluss */
  }
});
```

Listing 2: Ajax in Promise-Syntax

```
$.ajax ({
  url: "datei.php",
  type: "GET"
})
.done(function(data, textStatus, jqXHR) {
  /* Code */
})
.always( function(jqXHR, textStatus) {
  /* Code */
});
```

behandelt werden. Vorher war es hingegen so, dass Exceptions aufstiegen (*bubble*), sodass die weiteren Callbacks nicht mehr ausgeführt wurden. Sehen wir uns ein Beispiel dafür an:

```
var deferred = $.Deferred();
deferred.then(function() {
  console.log("1. Callback");
  throw new Error("Fehler im Callback");
})
.then(function() {
  console.log("2. Callback");
}, function(err) {
  console.log("Rejection Callback", err
    instanceof Error);
});
deferred.resolve();
```

In diesem Fall verwenden wir zuerst `.then()` mit einer Funktion, die einen Fehler auslöst. Beim zweiten `.then()` werden zwei Funktionen übergeben – die erste für den Fall, dass Deferred aufgelöst ist (*resolved*), und der zweite für den Fall, dass Deferred fehlgeschlagen ist (*rejected*).

In jQuery vor 3.0 erzeugt das folgende Ausgabe:

1. Callback

Es wird also nur die Meldung aus dem 1. Callback ausgegeben und danach die Ausführung des Skripts abgebrochen. In jQuery ab 3.0 erhält man hingegen folgende Ausgabe:

1. Callback
Rejection Callback true

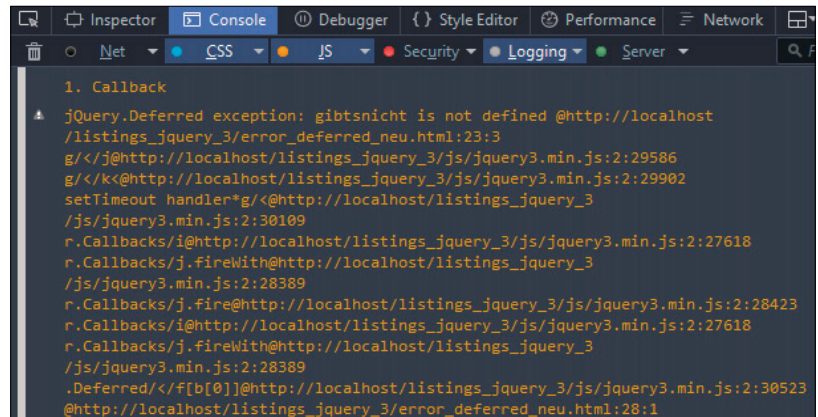
In diesem Fall wird beim zweiten `.then()` die zweite Funktion ausgeführt, eine Meldung ausgegeben, und `true` bestätigt, dass es sich beim Fehler um eine Instanz von `Error` handelt.

Wenn man nur einen Callback für den Fehler definieren möchte, kann man bei `.then()` statt der ersten Funktion `null` übergeben:

```
.then(null, function(err) {
```

Listing 3: Ajax-Verwendung – alte Art

```
$.ajax ({
  url: "datei.php",
  type: "GET"
})
.success(function(data, textStatus, jqXHR) {
  /* Code */
})
.complete( function(jqXHR, textStatus) {
  /* Code */
});
```



JavaScript-Fehler innerhalb von Deferreds werden in der Konsole geloggt (Bild 5)

```
console.log("Rejection Callback", err instanceof
  Error);
});
```

Alternativ können Sie auch das in jQuery 3.0 neu eingeführte `.catch()` einsetzen:

```
.catch(function(err) {
  console.log("Rejection Callback", err instanceof
    Error);
});
```

Im Beispiel haben wir einen Fehler über `throwError()` selbst erzeugt. Das demonstrierte Verhalten zeigt sich ebenfalls bei anderen Fehlern, etwa wenn man eine Funktion aufruft, die nicht existiert. Im folgenden Beispiel existiert die Funktion `gibtsnicht()` nicht:

```
var deferred = $.Deferred();
deferred.then(function() {
  console.log("1. Callback");
  gibtsnicht();
})
.then(function() {
  console.log("es geht weiter");
});
deferred.resolve();
console.log("außerhalb");
```

Der Code wird bei jQuery 3.0 weiter ausgeführt, und da es keinen Callback für den Fehlerfall und auch kein `.catch()` gibt, erscheint keine Fehlermeldung.

Im Gegensatz dazu wird bei jQuery < 3 das Skript abgebrochen und in der Konsole der Fehler `ReferenceError: gibtsnicht is not defined` ausgegeben.

Dieses neue Verhalten kann Probleme beim Debuggen machen. Hilfreich ist aber, dass standardmäßig Meldungen innerhalb der Konsole ausgegeben werden, wenn innerhalb eines Deferred eine JavaScript-Exception auftaucht (Bild 5). Falls Sie diese bei der Entwicklung nützlichen Konsolen-

Meldungen in der produktiven Version deaktivieren wollen, schreiben Sie:

```
jQuery.Deferred.exceptionHook = undefined;
```

Am häufigsten wird Ihnen diese andere Behandlung von Fehlern bei Ajax begegnen. Das unterschiedliche Verhalten von jQuery < 3.0 und jQuery 3.x zeigt sich deutlich in folgendem Beispiel:

```
$.ajax("datei.php")
  .then(function(data) {
    gibtsnicht();
  })
  .then(null, function(arg) {
    console.log("Im zweiten then");
  });
```

Bei jQuery < 3.0 wird die Abarbeitung aufgrund des Aufrufs der nicht definierten Funktion abgebrochen.

Bei jQuery 3.0 hingegen wird die zweite Meldung im zweiten *then* ausgegeben.

Wenn Sie das ursprüngliche Verhalten auch in jQuery 3.0 wiederhaben wollen, müssen Sie statt *.then()* auf *.done()* und *.fail()* zurückgreifen:

```
$.ajax("datei.php")
  .done(function(data) {
    gibtsnicht();
  })
  .fail(function() {
```

Links zum Thema

- Entwicklung von jQuery 3.0
<https://blog.jquery.com/2015/07/13/jquery-3-0-and-jquery-compat-3-0-alpha-versions-released>
<http://blog.jquery.com/2016/01/14/jquery-3-0-beta-released>
- Ende der Unterstützung für IE < 11
<https://www.microsoft.com/en-us/WindowsForBusiness/End-of-IE-support>
- Schnellere Selektoren in jQuery 3.0
<https://github.com/jquery/sizzle/issues/315#issuecomment-74336936>
- Angepasste Version von jQuery erstellen
<https://github.com/jquery/jquery/blob/master/README.md#how-to-build-your-own-jquery>
- Promises/A+
<https://promisesaplus.com>
- jQuery 3.0 Upgrade-Guide
<https://jquery.com/upgrade-guide/3.0/#callback-invocation>

```
C:\Windows\system32\cmd.exe
C:\Users\Florence\entwicklung\jquerycustom\jquery>grunt custom:-ajax
Running "custom:-ajax" (custom) task
Creating custom build...
? May jquery anonymously report usage statistics to improve the tool over time?
? May jquery anonymously report usage statistics to improve the tool over time?
yes
Running "build:all:*:-ajax" (build) task
- ajax
- ajax/jsonp
- ajax/load
- ajax/parseXML
- ajax/script
- ajax/var/location
- ajax/var/nonce
- ajax/var/rquery
- ajax/xhr
- manipulation/_evalUrl
- event/ajax
>> File 'dist/jquery.js' created.
Running "uglify:all" (uglify) task
File dist/jquery.min.map created (source map).
File dist/jquery.min.js created: 230.83 kB → 76.81 kB
>> 1 sourcemap created.
>> 1 file created.
Running "dist" task
Done.
C:\Users\Florence\entwicklung\jquerycustom\jquery>
```

Eigene Version: Sie können sich auch eine eigene Version von jQuery nur mit den benötigten Modulen erstellen (Bild 6)

```
console.log("in then");
});
```

Eine weitere Änderung ist, dass in jQuery 3.0 die Callback-Funktionen immer asynchron aufgerufen werden, auch wenn Deferred schon aufgelöst wurde:

```
var deferred = jQuery.Deferred();
deferred.resolve();
deferred.then(function() {
  console.log("Erfolgs-Callback");
});
console.log("Nach dem Binden");
```

Bei jQuery 3.0 ist die Ausgabe folglich:

```
Nach dem Binden
Erfolgs-Callback
```

In jQuery < 3.0 war die Reihenfolge umgekehrt. Ebenso gibt es eine Änderung beim *ready*-Handler, der jetzt immer asynchron aufgerufen wird.

```
$(function(){
  console.log("ready");
});

console.log("außerhalb von ready");
```

Bei jQuery 3.0 erscheint immer:

```
außerhalb von ready
ready
```

In Versionen vor jQuery 3.0 war die Ausgabe hingegen unterschiedlich, je nachdem, ob das Dokument *ready* war oder nicht.

Einige Änderungen betreffen die Ermittlung von Dimensionen von Elementen: So können `width()` und `height()` jetzt auch Nicht-Integer-Werte zurückgeben, also Fließkommazahlen. Rechnen Sie also damit, dass Sie bei jQuery 3.0 Werte wie 700.8333129882812 an Stellen erhalten, wo in der vorherigen Version nur 701 zurückgegeben wurde.

Ein weiterer Unterschied ist, dass bei `$(window).outerWidth()` die Scrollleisten mitberücksichtigt werden. Bei der Verwendung von `$(window).outerWidth()` erhalten Sie – sofern Scrollbalken vorhanden sind – bei jQuery 3.x Werte, die um 17px oder ähnlich größer sind. Dasselbe gilt auch für `$(window).outerHeight()`. Das ist an sich aber eine gute Idee, weil damit `$(window).outerWidth()` und `$(window).outerHeight()` auf denselben Werten operieren wie die CSS Media Queries.

Neu ist ebenfalls, dass `.offset()` nur noch in Fällen angewandt werden kann, wo es auch sinnvoll ist. `$(window).offset()` etwa erzeugt einen Fehler – vorher wurde in diesem Fall `undefined` zurückgegeben, aber der Code weiterhin ausgeführt.

Weitere Änderungen

Im CSS Object Model ist `CSS.escape()` vorgesehen, um Strings für die Verwendung in Selektoren zu maskieren. Das kann dann beispielsweise folgendermaßen aussehen:

```
var element = document.querySelector('#' + CSS.escape(id) + ' > span')
```

Dieses Feature wird allerdings derzeit nur in Chrome/Firefox unterstützt. Bei der Verwendung der neu eingeführten jQuery-Methode `$.escapeSelector` funktioniert es hingegen browserübergreifend. So können Sie auch ein Element mit der `id="abc.def"` auswählen (ansonsten würde ja der Selektor `abc.def` interpretiert als ein Element `abc` mit der Klasse `def`):

```
$( '#' + $.escapeSelector( "abc.def" )
```

Verändert hat sich außerdem das Verhalten bei `.show()/hide()/toggle()`, sodass weniger die berechneten Werte als die im Stylesheet spezifizierten Angaben herangezogen werden. Eine im Upgrade-Guide verlinkte Tabelle verrät alle Details.

Eine weitere Änderung betrifft `.data()`. Alle `data`-Namen werden nun in jQuery intern in der CamelCase-Schreibweise abgespeichert.

Wenn man bei jQuery vor 3.0 `.removeAttr()` bei einem booleschen Attribut wie `checked`, `selected` oder `readonly` verwendet, wird dieses Attribut auf falsch gesetzt – das passiert nun nicht mehr. Andererseits ist es in den meisten Fällen sowieso nicht empfehlenswert, `removeAttr("checked")` zu verwenden – `.prop("checked", false)` ist im Zweifelsfall die bessere Wahl. `.toggleClass()` ohne Argument aufzurufen ist in jQuery 3.0 deprecated – hier sollten Sie die Klasse angeben, die Sie wechselnd hinzufügen beziehungsweise entfernen lassen möchten.

Ebenfalls deprecated ist `$.parseJSON()`, da alle Browser, die jQuery 3.0 unterstützt, auch das native `JSON.parse()` im-

plementiert haben. Auf diese und weitere Spezialfälle weist das Migrate-Plug-in zuverlässig hin.

Man braucht nicht bei allen Projekten die gesamte Funktionalität, die jQuery beinhaltet. Deswegen können Sie sich eine angepasste Version von jQuery erstellen, die bestimmte Module ausschließt. So können Sie unter anderem folgende Module ausschließen:

- `ajax`: die Ajax-Funktionalität,
- `css`: die `.css()`-Methode und alle Module, die von `css` abhängen, das heißt, Effekte, Dimensionen et cetera,
- `deprecated`: die Methoden, die als veraltet/unerwünscht eingestuft werden, aber noch unterstützt werden,
- `effects`: Hierzu gehören `.animate()` und `.slideUp()` und Co.,
- `offset`: für Methoden wie `.offset()`, `.position()` et cetera,
- `deferred`: entfernt `$.Deferred` und `$.Callbacks`. Achtung, Module wie Ajax, die auf `$.Deferred` basieren, funktionieren dann ebenfalls nicht. Deshalb müssen Sie entweder eine eigene `$.Deferred`-Implementierung integrieren oder diese Module ebenfalls entfernen.

Eine vollständige Liste bietet die GitHub-Seite des Projekts. Bei der Erstellung der eigenen jQuery-Version können Sie dann angeben, welche Module ausgeschlossen werden sollen (Bild 6):

```
grunt custom:-ajax
```

Damit ist beispielsweise Ajax nicht dabei. Es lassen sich auch mehrere Module auf einmal ausschließen:

```
grunt custom:-ajax,-css,-deprecated,-dimensions,-effects,-event/alias,-offset,-wrap
```

Das war schon in vorherigen Versionen möglich. Für den, der es einfacher mag, gibt es mit jQuery 3.0 eine direkt zur Verfügung stehende Slim-Variante: Hier sind Ajax, Effekte und Code eingespart, der deprecated ist. Das Ergebnis: Slim ist mit 23,6 KByte gut 6 KByte leichter als die normale 3.0-Version. Diese Variante trägt Slim im Namen: <https://code.jquery.com/jquery-3.0.0.slim.min.js>.

Häufig wird kritisiert, dass jQuery immer automatisch zum Einsatz kommt, auch bei banalen Aufgaben, die sich rasch mit JavaScript pur erledigen lassen. Hilfreich ist hier, dass Sie sich einfach eine eigene Version zusammenstellen lassen können. In vielen Fällen kann auch schon die in jQuery 3.0 neu eingeführte Slim-Variante attraktiv sein. ■



Florence Maurice

ist Autorin, Trainerin und Programmiererin in München. Sie schreibt Bücher zu PHP und CSS3 und gibt Trainings per Video. Außerdem bloggt sie zu Webthemen unter:

<http://maurice-web.de/blog>

FUNKTIONALE PROGRAMMIERUNG IN JAVASCRIPT

Zentrale Rolle

Prinzipien und Techniken der funktionalen Programmierung in JavaScript.

Neben der objektorientierten Programmierung, die ich Ihnen in Ausgabe 7/2016 der **web & mobile developer** vorgestellt habe, ist die sogenannte funktionale Programmierung ein weiteres wichtiges Programmierparadigma, das in JavaScript eine zentrale Rolle spielt.

Doch was bedeutet funktionale Programmierung überhaupt? Welche Prinzipien liegen diesem Programmierparadigma zugrunde? Was ist der Unterschied zur objektorientierten Programmierung? Und was der Unterschied zur imperativen Programmierung?

Die Prinzipien der funktionalen Programmierung

Die funktionale Programmierung folgt im Wesentlichen vier Prinzipien, die im Folgenden kurz erläutert werden:

- **Prinzip 1:** Funktionen sind Objekte erster Klasse.
- **Prinzip 2:** Funktionen arbeiten mit unveränderlichen Datenstrukturen.
- **Prinzip 3:** Funktionen haben keine Nebeneffekte.
- **Prinzip 4:** Funktionale Programme sind deklarativ.

Funktionen sind Objekte erster Klasse (so genannte First Class Objects, manchmal auch als First Class Citizens bezeichnet). Funktionen können wie andere Objekte und primitive Werte ebenfalls Variablen zugewiesen werden, sie können als Argumente anderer Funktionen verwendet werden oder als deren Rückgabewert.

In nicht funktionalen Programmiersprachen dagegen (wie beispielsweise Java), beziehungsweise genauer gesagt in Sprachen, die keine funktionalen Konzepte unterstützen, werden Funktionen nicht als Objekte repräsentiert und können folglich auch nicht wie solche behandelt werden.

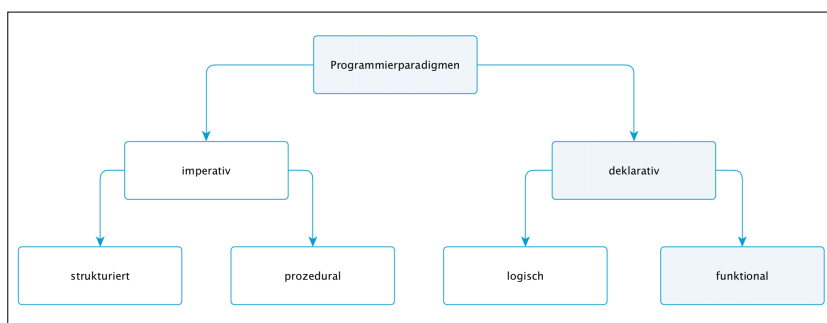
Die Datenstrukturen bei der funktionalen Programmierung sind in der Regel unveränderlich beziehungsweise werden nicht verändert. Vielmehr erzeugen Operationen, die auf Datenstrukturen durchgeführt werden, im Bedarfsfall neue Datenstrukturen und liefern diese als Ergebnis zurück. In rein funktionalen Programmiersprachen können beispielsweise Listen oder andere Datenstrukturen, die einmal angelegt worden sind, nachträglich nicht mehr geändert werden (beziehungsweise nur über Umwege).

Da JavaScript keine rein funktionale Programmiersprache ist, können hier allerdings sehr wohl Arrays (die in der Regel als Datenstruktur bei der funktionalen Programmierung in JavaScript zugrunde liegen) jederzeit verändert werden.

Hinzu kommt, dass bei der funktionalen Programmierung die Funktionen überhaupt keine Nebeneffekte haben und sich eher wie mathematische Funktionen verhalten sollten. Funktionen liefern also bei gleichen Eingaben immer das gleiche Ergebnis, lösen dabei aber keinerlei Nebeneffekte aus. In rein funktionalen Sprachen werden Nebeneffekte bereits durch die Sprache selbst verhindert. JavaScript als nicht rein funktionale Programmiersprache erlaubt es dagegen durchaus, dass Funktionen bei gleichen Eingaben sowohl unterschiedliche Ergebnisse liefern als auch Nebeneffekte haben können.

Funktionale Programme sind deklarativ (Bild 1). Man formuliert sein Programm also eher so, dass man sagt, was gemacht werden soll, und nicht, wie etwas gemacht werden soll. Dadurch sind funktionale Programme gegenüber dem äquivalenten imperativen Code in der Regel besser lesbar, sprechender und kompakter.

Im Unterschied zur objektorientierten Programmierung liegt der Fokus bei der funktionalen Programmierung auf



Einordnung der funktionalen Programmierung (Bild 1)

Funktionen, nicht auf Objekten. JavaScript vereint diese beiden Programmierparadigmen. Beispielsweise können Sie damit Ihr Programm objektorientiert strukturieren, das heißt, mit Objekten arbeiten, und innerhalb von Objektmethoden wiederum funktional programmieren – statt beispielsweise imperativ.

JavaScript bietet für Arrays bereits eine Reihe funktionaler Methoden an, mit denen sich, wie im Folgenden gezeigt, gewisse Problemstellungen einfacher lösen lassen als mit der imperativen Programmierung.

Über Arrays iterieren

Wenn Sie beispielsweise über ein Array iterieren wollen, haben Sie gleich mehrere (imperative) Möglichkeiten:

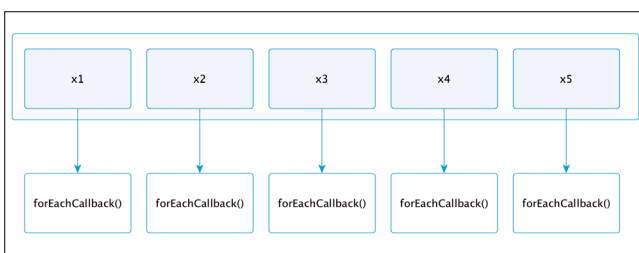
Listing 1: Objektmodell in JSON

```
'use strict';
let persons = [
  {
    "firstName": "Max",
    "lastName": "Mustermann",
    "contacts": [
      {
        "type": "Festnetz",
        "value": "2345/23452345"
      },
      {
        "type": "Mobilnummer",
        "value": "0124/23452345"
      },
      {
        "type": "Mobilnummer",
        "value": "0123/23452345"
      },
      {
        "type": "E-Mail",
        "value": "max.mustermann@example.com"
      }
    ]
  },
  {
    "firstName": "Moritz",
    "lastName": "Mustermann",
    "contacts": [
      {
        "type": "Festnetz",
        "value": "2345/56565656"
      },
      {
        "type": "Mobilnummer",
        "value": "0123/56565656"
      },
      {
        "type": "E-Mail",
        "value": "moritz.mustermann@example.com"
      }
    ]
  }
]
```

- über eine Zählerschleife (*for*-Schleife),
- über eine kopfgesteuerte Schleife (*while*-Schleife),
- über eine fußgesteuerte Schleife (*do-while*-Schleife),
- über eine *for-in*-Schleife,
- über eine *for-of*-Schleife.

Ein Beispiel soll auf Basis des Objektmodells (Listing 1) zeigen, wie die Elemente eines Arrays mit Hilfe einer Zählerschleife ausgegeben werden können. Das Array, über das hier iteriert wird, enthält verschiedene Personen-Objekte, bestehend aus Vorname, Nachname und einer Liste von Kontaktinformationen. Der folgende Code soll lediglich die Vornamen der im Array enthaltenen Personen-Objekte ausgeben:

```
'use strict';
...
for (let i = 0; i < persons.length; i++) {
  console.log(persons[i].firstName);
}
```



forEach(): Veranschaulichung der Methode *forEach()* (Bild 2)

Das ist das, was passieren soll. Dabei beschäftigt sich der Code eigentlich aber viel zu sehr damit, wie das Ganze vonstattengehen soll: Zählervariable initialisieren, Abbruchbedingung überprüfen, indexbasierter Zugriff auf das Array und anschließendes Hochzählen der Zählervariable.

Imperative Programme enthalten also, wie man sieht, viel zusätzlichen Code, der das Programm nicht sehr leserlich macht, sondern eher unnötig aufbläht (Boilerplate-Code). Sicher hat man als Entwickler irgendwann einen Blick für solche imperativen Kontrollstrukturen und weiß in einfachen Fällen auf Anhieb, was eine Zählerschleife im konkreten Fall macht. In der Praxis verhält es sich aber nicht immer so einfach wie im gezeigten Beispiel: Häufig werden Schleifen geschachtelt, innerhalb der Schleifen findet man dann bedingte Anweisungen, Verzweigungen et cetera, sodass der Überblick schnell verloren geht.

Mit Hilfe der funktionalen Programmierung lassen sich Problemstellungen in den meisten Fällen dagegen viel lesbarer formulieren. Für die gegebene Problemstellung eignet sich beispielsweise die Methode *forEach()* viel besser:

```
'use strict';
...
persons.forEach((person, index, array) => {
  console.log(person.firstName);
});
```

Als Argument übergibt man dieser Methode eine Funktion (Callback-Funktion), die dann für jedes Element im Array mit drei Argumenten aufgerufen wird: dem jeweiligen Ele- ►

ment, dem Index des Elements und schließlich dem Array selbst (Bild 2).

Auch wenn der Code bezogen auf die Anzahl der Zeilen nicht unbedingt kürzer wird, ist er doch schon um einiges besser lesbar. Im Gegensatz zur imperativen Variante liegt der Fokus nämlich jetzt auf der Logik des Programms, nicht auf der Schleife an sich. Übergibt man anstelle einer anonymen Funktion eine benannte Funktion, lässt sich das Programm fast wie natürliche Sprache lesen:

```
'use strict';
function printFirstName(person) {
    console.log(person.firstName);
}
persons.forEach(printFirstName);
```

Oft ist es so, dass man nicht nur über die Elemente eines Arrays iterieren möchte, sondern zeitgleich auch für jedes Element einen Wert ermitteln und diesen in einem anderen Array speichern will. Beispielsweise, um für ein Array natürlicher Zahlen zu jeder Zahl das Quadrat zu berechnen, oder um für ein Array von Objekten von jedem Objekt eine Eigenschaft auszulesen und den entsprechenden Wert in ein neues Array zu kopieren.

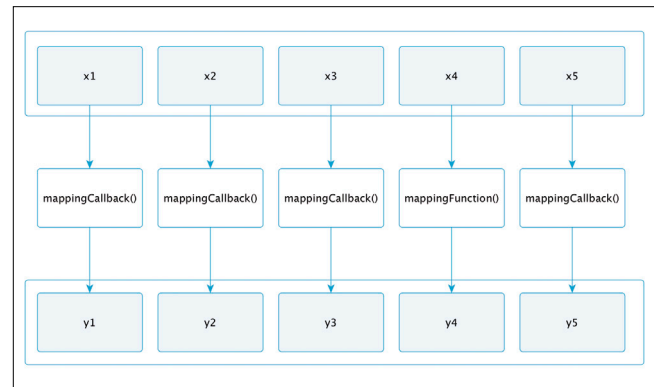
Nehmen wir als Beispiel dazu das bekannte Objektmodell und nehmen an, dass aus dem dort gezeigten Array mit den Personen-Objekten die Vornamen der Personen in ein neues Array kopiert werden sollen. Imperativ würde man diese Problemstellung vermutlich so lösen:

```
'use strict';
let firstNames = [];
for(let i=0; i<persons.length; i++) {
    firstNames.push(persons[i].firstName);
}
console.log(firstNames);
```

Wie schon im Beispiel für die Iteration nimmt auch in diesem Code die *for*-Schleife einen großen Teil des Codes in Anspruch, obwohl sie nur Mittel zum Zweck ist. Mit der Methode *map()* dagegen wird der Code deutlich sprechender:

```
'use strict';
let firstNames = persons.map((person, index, array) => {
    return person.firstName;
});
console.log(firstNames);
function getFirstName(person) {
    return person.firstName;
}
let firstNames = persons.map(getFirstName);
```

Als Argument erwartet diese Methode, ähnlich wie *forEach()*, eine Funktion, die dann für jedes Element im Array aufgerufen wird. Der Rückgabewert dieser Funktion bestimmt dabei den Wert, der für das jeweilige Element in das Ziel-Array geschrieben werden soll (Bild 3).



map(): Veranschaulichung der Methode *map()* (Bild 3)

Ein weiterer häufig anzutreffender Anwendungsfall ist das Filtern von Elementen in einem Array. Angenommen, man möchte alle Mobilnummern eines Personen-Objekts herausfiltern. Imperativ würde man diese Aufgabe vermutlich unter Verwendung einer Zählerschleife implementieren:

```
'use strict';
let maxContacts = persons[0].contacts;
let maxMobileNumbers = [];
for(let i=0; i<maxContacts.length; i++) {
    if(maxContacts[i].type === 'Mobilnummer') {
        maxMobileNumbers.push(maxContacts[i]);
    }
}
console.log(maxMobileNumbers);
```

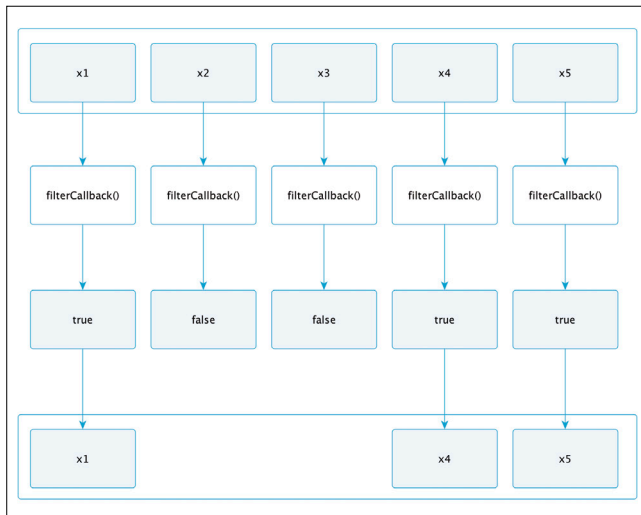
Für jede Kontaktinformation in dem Array, das der Eigenschaft *contact* hinterlegt ist, wird hier innerhalb der Schleife geprüft, ob der Wert für den Typen der Kontaktinformation (Eigenschaft *type*) gleich dem Wert *Mobilnummer* ist. Ist dies der Fall, wird das Element dem Ergebnis-Array *maxMobileNumbers* hinzugefügt.

Werte eines Arrays filtern

Mit der Methode *filter()* ist dies deutlich einfacher. Als Argument übergibt man auch hier, wie schon bei *forEach()* und *map()*, eine Funktion, die für jedes Element im Array aufgerufen wird:

```
'use strict';
let maxContacts = persons[0].contacts;
let maxMobileNumbers = maxContacts.filter(
    (contact, index, array) => {
        return contact.type === 'Mobilnummer';
    }
);
console.log(maxMobileNumbers);
```

Der Rückgabewert dieser Funktion bestimmt in diesem Fall, ob ein Element in das neue Array übernommen wird: Gibt die Funktion ein *true* zurück, wird es in das neue Array übernommen, andernfalls nicht (Bild 4). Innerhalb der Callback-Funk-



filter(): Veranschaulichung der Methode *filter()* (Bild 4)

tion hat man dabei wieder Zugriff auf das aktuelle Element, dessen Index sowie auf das gesamte Array.

Eine weitere bekannte Methode im Bunde der funktionalen Methoden für Arrays ist die Methode *reduce()*. Diese Methode dient dazu, ausgehend von den Elementen eines Arrays einen einzigen repräsentativen Wert zu ermitteln, quasi die Elemente eines Arrays zu einem einzelnen Wert zu reduzieren.

Arrays zu einem Wert reduzieren

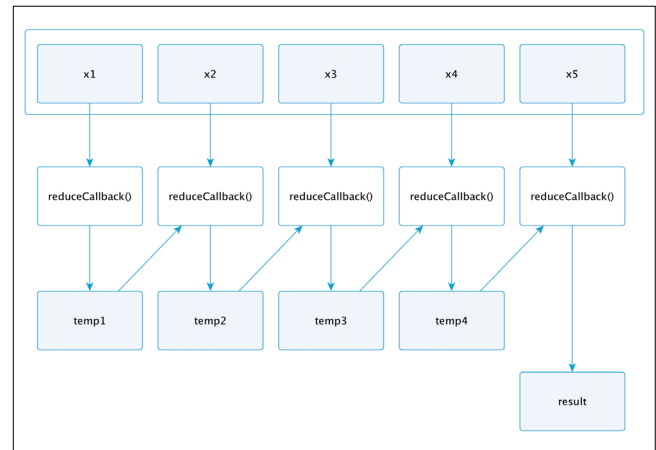
Zur Veranschaulichung dieser Methode nehmen wir an, dass auf Basis des Objektmodells des Personen-Arrays die Anzahl aller Kontaktinformationen ermittelt werden soll. Auf imperativem Wege würde man diese Problemstellung wahrscheinlich über eine Zählerschleife lösen.

Über die Methode *reduce()* geht das wieder einfacher. Als Argument übergibt man eine Funktion, die wie gewohnt für jedes Element im Array aufgerufen wird.

Wie bei den anderen besprochenen Methoden hat man innerhalb dieser übergebenen Funktion Zugriff auf Element, Index und das gesamte Array. Zusätzlich bekommt die Funktion aber noch den aktuell akkumulierten Wert der vorigen Iteration als Argument übergeben, wobei sich der Startwert optional als zweites Argument der Methode *reduce()* übergeben lässt (Bild 5).

Für jedes Element im Personen-Array wird nun die übergebene Funktion aufgerufen und die Anzahl der Kontaktinformationen der jeweiligen Person auf die Gesamtanzahl aller Kontaktinformationen (*totalContacts*) addiert:

```
'use strict';
let totalContacts = persons.reduce(
  (previousValue, person, index, array) => {
    return previousValue + person.contacts.length;
  },
  0
);
console.log(totalContacts); // 7
```



reduce(): Veranschaulichung der Methode *reduce()* (Bild 5)

Man sieht: Bereits die gezeigten standardmäßig zur Verfügung stehenden Methoden für Arrays machen den Quelltext um einiges besser lesbar sowie die Lösung der jeweiligen Problemstellung viel eleganter. Der wahre Vorteil zeigt sich jedoch erst, wenn man die Methoden miteinander kombiniert und die Aufrufe verbindet, etwa wie im folgenden Listing:

```
'use strict';
persons
  .filter(person => person.age > 45)
  .map(person => person.contacts)
  .reduce((allContacts, contacts) =>
    allContacts.concat(contacts), [])
  .filter(contact => contact.type === 'Festnetz')
  .forEach(contact => console.log(contact.value));
```

Hier werden zuerst die Personen herausgefiltert, die älter als 45 sind. Dann werden von diesen Personen die Kontaktinformationen herausgemappt, dann daraus die Festnetznummern herausgefiltert und anschließend darüber iteriert. Der Code, der das erreicht, ist in der funktionalen Variante extrem gut lesbar, vor allem wenn man die einzelnen Callback-Funktionen nicht anonym, sondern mit Namen definiert:

```
'use strict';
persons
  .filter(personOver45)
  .map(getContacts)
  .reduce(mergeContacts)
  .filter(isFixedLine)
  .forEach(print);
```

Es gibt seit ES2015 noch einige weitere Methoden von Arrays, die in diesem Zusammenhang erwähnenswert sind, an dieser Stelle aber nicht im Detail besprochen werden sollen:

- Über die Methode *every()* kann geprüft werden, ob jedes Element in einem Array ein durch die übergebene Callback-Funktion definiertes Kriterium erfüllt.
- Über die Methode *some()* kann geprüft werden, ob mindestens ein Element in einem Array ein durch die überge-

bene Callback-Funktion definiertes Kriterium erfüllt.

- Die Methode *reduceRight()* funktioniert vom Prinzip her wie die Methode *reduce()*, arbeitet aber die Elemente nicht von links nach rechts, sondern von rechts nach links ab. Eine Übersicht aller funktionaler Methoden von Arrays zeigt die [Tabelle 1](#).

Funktionale Techniken und Entwurfsmuster

Neben diesen grundlegenden Methoden, die in JavaScript für Arrays bereitstehen, gibt es natürlich noch eine Reihe weiterer funktionaler Techniken und Entwurfsmuster, etwa Closures, Memoization, partielle Auswertung und Currying.

Eine relativ bekannte Technik der funktionalen Programmierung sind sogenannte Closures. Diese kommen bei einer besonderen Art von Funktion höherer Ordnung zustande, nämlich dann, wenn eine Funktion eine andere Funktion zurückliefert, die wiederum auf die Variablen beziehungsweise Parameter der äußeren Funktion zugreift.

Was sie dabei so besonders macht, ist die Tatsache, dass die Variablen (der ursprünglich äußeren Funktion) auch noch zur Verfügung stehen, wenn die äußere Funktion bereits beendet wurde. Berücksichtigt wird hierbei also jeweils die aktuelle Belegung der Variablen.

Die zurückgegebene Funktion schließt die Variablen sozusagen ein – daher der Name Closure. Das bedeutet, dass sich eine äußere Funktion durchaus mehrmals aufrufen lässt und verschiedene Closures, ausgehend von der aktuellen Umgebung mit unterschiedlichen Variablenbelegungen, zurückliefert. Ein Beispiel dazu zeigt das folgende Listing:

```
'use strict';
function counterFactory(name) {
  let i=0;
  return function() {
    i++;
    console.log(name + ', ' + i);
  }
}
```

Tabelle 1: Funktionale Methoden von Arrays

Methode	Beschreibung
<i>every()</i>	Prüft, ob alle Elemente im Array ein bestimmtes Kriterium erfüllen.
<i>filter()</i>	Filtern von Elementen eines Arrays, die ein bestimmtes Kriterium erfüllen.
<i>forEach()</i>	Iteration über alle Elemente eines Arrays.
<i>map()</i>	Elemente eines Arrays auf einen jeweils anderen Wert abbilden.
<i>reduce()</i>	Zusammenfassen der Elemente eines Arrays zu einem einzelnen Wert.
<i>reduce-Right()</i>	Wie <i>reduce()</i> , nur dass die Elemente im Array von rechts nach links durchgegangen werden.
<i>some()</i>	Prüft, ob ein oder mehrere Elemente im Array ein bestimmtes Kriterium erfüllen.

Listing 2: Fibonacci-Zahlen

```
'use strict';
let fibonacciWithCache = function() {
  let cache = [0, 1];
  let fibonacci = function(n) {
    let result = cache[n];
    if (typeof result !== 'number') {
      console.log('Neuberechnung für: ', n);
      result = fibonacci(n - 1) + fibonacci(n - 2);
      cache[n] = result;
    }
    return result;
  };
  return fibonacci;
};
let fibonacci = fibonacciWithCache();
console.log('Ergebnis: ', fibonacci(11));
console.log('Ergebnis: ', fibonacci(11));
console.log('Ergebnis: ', fibonacci(15));
```

```
let counter1 = counterFactory('Zähler 1');
counter1(); // Zähler 1: 1
counter1(); // Zähler 1: 2
let counter2 = counterFactory('Zähler 2');
counter1(); // Zähler 2: 1
counter2(); // Zähler 2: 2
```

Hier kommen zwei dieser Konstrukte zum Einsatz: *counter1* und *counter2*. Jede dieser über die Funktion *counterFactory()* erzeugten Funktionen hat nur Zugriff auf die Umgebung, in der sie erstellt wurde. Ein Aufruf von *counter1()* beziehungsweise *counter2()* verändert nicht die Zählervariable *i* des jeweils anderen Zählers.

Ein weiterer Vorteil: Von außen lässt sich die Variable *i* nicht ändern. Eine wichtige Grundlage für die Datenkapselung, weshalb Closures einen hohen Stellenwert in der fortgeschrittenen JavaScript-Entwicklung haben und in vielen Entwurfsmustern Anwendung finden, wie beispielsweise dem Memoization-Entwurfsmuster.

Memoization

Ein bekanntes Beispiel für die Anwendung einer Closure ist die Implementierung eines Caching-Mechanismus, auch Memoization-Entwurfsmuster genannt. Der Code in [Listing 2](#) zur Berechnung von Fibonacci-Zahlen ist (leicht abgeändert) auch in Douglas Crockfords lesenswertem Klassiker »JavaScript – The Good Parts« zu finden. Das Beispiel dort ist zwar noch etwas eleganter, weil es zusätzlich die IIFE-Technik (Immediately Invoked Function Expressions) anwendet. Der wesentliche Bestandteil des Musters, die Closure, ist aber der Gleiche.

Was passiert hier? Die Funktion *fibonacciWithCache()* verwaltet ein Array von Zahlen (den Cache), erzeugt eine Funktion (*fibonacci()*) und gibt diese zurück. Ruft man nun diese

Listing 3: Imperative Programmierung

```
'use strict';
function createPerson(firstName, lastName, age) {
  return {
    firstName: firstName,
    lastName: lastName,
    age: age
  }
}
let firstName = 'Max';
let maxMustermann = createPerson(
  firstName,
  'Mustermann',
  44
);
let maxMueller = createPerson(
  firstName,
  'Müller',
  47
);
let maxMeier = createPerson(
  firstName,
  'Meier',
  55
);
console.log(maxMustermann);
console.log(maxMueller);
console.log(maxMeier);
```

Listing 4: Wiederverwendung von Funktionen

```
'use strict';
function createPerson(firstName, lastName, age) {
  return {
    firstName: firstName,
    lastName: lastName,
    age: age
  }
}
function createPersonWithFirstNameMax(lastName, age) {
  return createPerson('Max', lastName, age);
}
let maxMustermann = createPersonWithFirstNameMax(
  'Mustermann',
  44
);
let maxMueller = createPersonWithFirstNameMax(
  'Müller',
  47
);
let maxMeier = createPersonWithFirstNameMax(
  'Meier',
  55
);
console.log(maxMustermann);
console.log(maxMueller);
console.log(maxMeier);
```

Funktion erstmals mit einem bestimmten Zahlenwert auf (zum Beispiel *fibonacci(11)*), so werden die entsprechenden Fibonacci-Zahlen rekursiv bis zu diesem Wert ermittelt und das Ergebnis wird zurückgegeben. So weit noch nichts Besonderes. Zusätzlich wird aber jede ermittelte Fibonacci-Zahl nach ihrer Berechnung auch in den Cache geschrieben, so dass nachfolgende Aufrufe der Funktion *fibonacci()* die entsprechenden Zahlen nur dann neu berechnen, wenn sie sich noch nicht im Cache befinden.

Partielle Auswertung

Hin und wieder kann es der Fall sein, dass man eine Funktion mehrmals mit den gleichen beziehungsweise zu Teilen gleichen Parameterwerten aufrufen möchte. Bei der imperativen Programmierung ist es in solchen Fällen üblich, die wieder verwendeten Werte in entsprechenden Variablen zu speichern und diesen Variablen dann der jeweiligen Funktion als Parameter zu übergeben (Listing 3).

Bei der funktionalen Programmierung geht das dank der sogenannten partiellen Auswertung (beziehungsweise Partial Application) einfacher. Die Idee dabei ist, eine Funktion zunächst mit den gleichbleibenden Parametern auszuwerten (diese Parameter werden dabei gebunden) und eine neue Funktion zu erstellen, die nur noch die verbleibenden Parameter (das heißt, die ungebundenen Parameter) erwartet.

Um dieses Prinzip besser zu verstehen, ist es am besten, zunächst zu verstehen, welche gedankliche Zwischenschritte der partiellen Auswertung vorausgehen. Der erste Schritt ist dabei leicht nachvollziehbar, denn so würde man auch bei der imperativen Programmierung vorgehen: Man definiert eine neue, speziellere Funktion, welche die alte Funktion mit den vorbelegten Parametern aufruft (Listing 4).

Der Nachteil hiervon ist natürlich, dass neue Funktionen immer einzeln und händisch definiert werden müssen. Für jeden vorbelegten Wert von *firstName* müsste eine neue Funktion deklariert werden:

```
'use strict';
...
function createPersonWithFirstNameMoritz(lastName, age) {
  return createPerson('Moritz', lastName, age);
}
```

Das geht besser, wie im Folgenden zu sehen. Und zwar kommen die funktionalen Aspekte von JavaScript zu Hilfe, denn dank derer ist es möglich, eine Funktion zu erstellen, die dynamisch eine andere Funktion zurückgibt. Warum also nicht eine Funktion erstellen, die für eine beliebige Belegung von *firstName* eine entsprechende Funktion zurückgibt, in der ►

firstName belegt ist und in der *createPerson()* mit dem entsprechenden Wert aufgerufen wird? Wie das geht, zeigt das [Listing 5](#), wobei die aus dem Vorangehenden bekannten Closures zum Einsatz kommen.

Dies ist schon besser und deckt alle Fälle ab, in denen der Wert von *firstName* feststeht. Was aber, wenn nicht nur *firstName*, sondern auch *lastName* vorbelegt werden soll? In diesem Fall funktioniert das oben gezeigte Vorgehen nicht mehr. Der nächste Schritt wäre also, die entsprechende Funktion so generisch zu machen, dass sie mit beliebigen vorgegebenen Parametern zurechtkommt.

Die Implementierung einer solchen Funktion ist eigentlich relativ einfach ([Listing 6](#)). Was man möchte, ist ja, dass ein Teil der Parameter in einer Closure gebunden wird und der Rest der Parameter ungebunden bleibt.

Man könnte auch von zwei Arrays sprechen: ein Array von Parametern, die beim Aufruf der äußeren Funktion gebunden werden, sowie ein Array von Parametern, die erst beim Aufruf der inneren Funktion gebunden werden (und solange ungebunden bleiben).

Beide Arrays ergeben sich aus den *arguments*-Objekten der äußeren und der inneren Funktion. Um diese Objekte jeweils in ein Array umzuwandeln, kommt die Technik des Methodenborgens zum Einsatz (mit ES2015 beziehungsweise Rest-Parametern lässt sich das Ganze wie später gezeigt noch weiter vereinfachen).

Auf diese Weise erhält man zwei Arrays: *parameterBound*, das beim Aufruf der äußeren Funktion (*createPersonFactory('Max')*) erzeugt wird, und *parameterUnbound*, das erst beim Aufruf der inneren Funktion erzeugt wird. Letzterer Aufruf führt auch dazu, dass beide Arrays zum Array *allParameters* zusammengefasst werden und mit diesen Parametern dann die Funktion *createPerson()* aufgerufen wird.

Wenn man sich schließlich die Funktion *createPersonFactory()* genau ansieht, fällt auf, dass nur an einer Stelle noch ein Bezug zu der Funktion *createPerson()* besteht, nämlich genau dann, wenn diese Funktion über *apply()* aufgerufen wird. Warum aber nicht auch das noch auslagern und die

Listing 6: Neuer Wert von firstName

```
'use strict';
...
function createPersonFactory() {
  let parameterBound = Array.prototype
    .slice.call(arguments, 0);
  return function() {
    let parameterUnbound = Array.prototype
      .slice.call(arguments, 0);
    let allParameters = parameterBound
      .concat(parameterUnbound);
    return createPerson
      .apply(this, allParameters);
  };
}

let createPersonWithFirstNameMax =
  createPersonFactory('Max');
let createPersonWithFirstNameMoritz =
  createPersonFactory('Moritz');
let maxMustermann =
  createPersonWithFirstNameMax('Mustermann', 44);
let maxMueller =
  createPersonWithFirstNameMax('Müller', 47);
let maxMeier =
  createPersonWithFirstNameMax('Meier', 55);
```

Funktion, die aufgerufen werden soll, allgemein halten und als Parameter übergeben?

[Listing 7](#) zeigt die generische Funktion *partial()*, die für beliebige Funktionen eine beliebige Anzahl an Parametern entgegennimmt und eine Funktion zurückgibt, in der diese Parameter gebunden, die restlichen Parameter jedoch ungebunden sind.

Der erste Parameter dieser Funktion ist diesmal die Funktion, die partiell ausgewertet werden soll. Alle weiteren Pa-

Listing 5: Generische partielle Auswertung

```
'use strict';
function createPersonWithFirstName(firstName) {
  return function(lastName, age) {
    return createPerson(firstName, lastName, age);
  }
}

let createPersonWithFirstNameMax =
  createPersonWithFirstName('Max');
let createPersonWithFirstNameMoritz =
  createPersonWithFirstName('Moritz');
let maxMustermann =
  createPersonWithFirstNameMax('Mustermann', 44);
let maxMueller =
  createPersonWithFirstNameMax('Müller', 47);

let maxMeier =
  createPersonWithFirstNameMax('Meier', 55);
console.log(maxMustermann);
console.log(maxMueller);
console.log(maxMeier);
let moritzMustermann =
  createPersonWithFirstNameMoritz('Mustermann', 25);
let moritzMueller =
  createPersonWithFirstNameMoritz('Müller', 26);
let moritzMeier =
  createPersonWithFirstNameMoritz('Meier', 27);
console.log(moritzMustermann);
console.log(moritzMueller);
console.log(moritzMeier);
```


Listing 7: Generische Funktions-Fabrik

```
'use strict';
...
function partial(fn /*, parameter...*/) {
  let parameterBound = Array.prototype
    .slice.call(arguments, 1);
  return function() {
    let parameterUnbound = Array.prototype
      .slice.call(arguments, 0);
    return fn.apply(this, parameterBound
      .concat(parameterUnbound));
  };
}
let createPersonWithFirstNameMax =
  partial(createPerson, 'Max');
let createPersonWithFirstNameMoritz =
  partial(createPerson, 'Moritz');
let maxMustermann =
  createPersonWithFirstNameMax('Mustermann', 44);
let maxMueller =
  createPersonWithFirstNameMax('Müller', 47);
let maxMeier =
  createPersonWithFirstNameMax('Meier', 55);
let createMaxMustermann =
  partial(createPerson, 'Max', 'Mustermann');
let maxMustermann2 = createMaxMustermann(55);
```

parameter werden weiterhin nicht explizit angegeben, beim Umwandeln des *arguments*-Objekts in das *parameterBound*-Array müssen Sie aber diesmal *slice()* ab Index 1 anwenden, also hinter dem Parameter, der das Funktionsobjekt enthält.

In ES2015 ist die Implementierung dank REST-Parameter und Spread-Operator sogar noch eleganter:

```
'use strict';
function partial(fn, ...parameterBound) {
  return function (...parameterUnbound) {
    return fn(...parameterBound, ...parameterUnbound);
  };
}
```

Alle gezeigten Implementierungen von *partial()* haben jedoch eine Einschränkung: Es besteht lediglich die Möglichkeit, Parameter von links beginnend zu binden. Das heißt beispielsweise, dass mit *partial()* keine Variante auf Basis von *createPerson()* erzeugt werden, in der nur der Parameter *age* gebunden ist. Hierzu müssten die Parameter von rechts beginnend gebunden werden. Des Weiteren ist es mit den bisher gezeigten Implementierungen nicht möglich, irgendeinen beliebigen Parameter mittendrin zu binden, beispielsweise für *createPerson()* den Parameter *lastName*.

Im Folgenden seien daher zwei Varianten von *partial()* vorgestellt: die partielle Auswertung von rechts ausgehend sowie die partielle Auswertung mit Platzhaltern.

Listing 8: Generische Funktions-Fabrik

```
'use strict';
function partialRight(fn /*, parameter...*/) {
  let parameterBound = Array.prototype
    .slice.call(arguments, 1);
  return function() {
    let parameterUnbound = Array.prototype
      .slice.call(arguments);
    return fn.apply(this, parameterUnbound
      .concat(parameterBound));
  };
}
let createPersonWithAge44 =
  partialRight(createPerson, 44);
let createPersonWithAge55 =
  partialRight(createPerson, 55);

let maxMustermann = createPersonWithAge44(
  'Max',
  'Mustermann'
);
let moritzMustermann = createPersonWithAge55(
  'Moritz',
  'Mustermann'
);
```

Die eben gezeigte generische Funktion wird auch *partial-Left()* genannt. Analog dazu gibt es die Funktion *partial-Right()*, bei der die Parameter von rechts beginnend ausgewertet werden. Das Einzige, was hierfür an der bisherigen Implementierung geändert werden muss, ist die Reihenfolge, in der die beiden Parameter-Arrays miteinander konkateniert werden (Listing 8). Die ES2015-Variante sieht so aus:

```
'use strict';
function partialRight(fn, ...parameterBound) {
  return function (...parameterUnbound) {
    return fn(...parameterUnbound, ...parameterBound);
  };
}
```

Wenn eine Funktion hinsichtlich beliebiger Parameter partiell ausgewertet werden soll, funktionieren die bisherigen Lösungen nicht mehr: *partial()* beziehungsweise *partialLeft()* wertet die Parameter von links aus, *partialRight()* von rechts. Um beliebige Parameter zu erlauben, muss man einige Erweiterungen durchführen.

Das Prinzip dabei ist, mit einem bestimmten Platzhalter-Wert zu arbeiten, dann innerhalb der *partial()*-Funktion zu prüfen, ob ein übergebener Parameter diesem Platzhalter-Wert entspricht, und abhängig davon das Parameter-Array zu bilden (Listing 9).

Die wesentlichen Änderungen spielen sich in der inneren Funktion ab. Hier wird zunächst ein neues Array erstellt, in dem alle konkreten Parameter gesammelt werden. Dazu ►

Listing 9: Partielle Auswertung mit Platzhaltern

```
'use strict';
let _ = {}; // Platzhalter
function partialWithPlaceholders(f /*, parameter...*/)
{
    let parameterBound = Array.prototype
        .slice.call(arguments, 1);
    return function() {
        let i,
            parameter = [],
            parameterUnbound = Array.prototype
                .slice.call(arguments, 0);
        for(i=0; i<parameterBound.length; i++) {
            if(parameterBound[i] !== _) {
                parameter[i] = parameterBound[i];
            } else {
                parameter[i] = parameterUnbound.shift();
            }
        }
        return f.apply(this, parameter
            .concat(parameterUnbound));
    };
}

let createPersonWithLastNameMustermann =
    partialWithPlaceholders(createPerson, _,
        'Mustermann', _);
let maxMustermann =
    createPersonWithLastNameMustermann('Max', 44);
let moritzMustermann =
    createPersonWithLastNameMustermann('Moritz', 55);
console.log(maxMustermann);
console.log(moritzMustermann);
```

wird über das Array gebundener Parameter iteriert. Wenn es sich bei einem Parameter nicht um den Platzhalter handelt, wird der Parameter direkt in das Zielarray übernommen. Für den Fall dagegen, dass es sich bei dem Parameter um den Platzhalter-Wert handelt, wird der Parameter aus dem Array *parameterUnbound* verwendet.

Hierbei wird die Methode *shift()* aufgerufen, die das erste Element aus einem Array löscht sowie gleichzeitig zurückgibt. Übrig bleiben auf diese Weise alle hinten stehenden Parameter, die bei der partiellen Auswertung überhaupt nicht übergeben wurden (auch nicht als Platzhalter).

Currying

Unter dem Begriff Currying versteht man eine Technik, bei der eine Funktion mit mehreren Parametern in mehrere Funktionen mit jeweils einem Parameter umgewandelt wird. Der verkettete Aufruf dieser einparametrischen Funktionen führt dann zu dem gleichen Ergebnis wie der Aufruf der einzelnen mehrparametrischen Funktion.

Nehmen wir als Beispiel die bereits bekannte Funktion *createPerson()*, eine Funktion mit drei Parametern: *firstName*, *lastName* und *age*. Die Curry-Variante dieser Funktion gibt eine Funktion zurück (und schließt *firstName* in einer Closure ein), die wiederum eine Funktion zurückgibt (die *lastName* in einer Closure einschließt), die erneut eine Funktion zurückgibt.

Doch JavaScript wäre nicht JavaScript, wenn man nicht auch hier eine generische Funktion implementieren könnte, die zu beliebigen Funktionen die äquivalente Curry-Variante erzeugt.

Fazit

Die funktionale Programmierung ist ein mächtiges Programmierparadigma, mit dessen Hilfe sich gegenüber der imperativen Programmierung viele Problemstellungen einfacher lösen lassen. JavaScript ist zwar keine rein funktionale Pro-

Tabelle 2: Bibliotheken und Anlaufstellen

Bezeichnung	URL
Awesome FP JS	https://github.com/stoeffel/awesome-fp-js
101	https://github.com/tjmehta/101
Folktale	http://folktalejs.org
functional.js	http://functionaljs.com
immutable-js	https://github.com/facebook/immutable-js
Lodash	https://lodash.com
Lodash FP	https://github.com/lodash/lodash/wiki/FP-Guide
Ramda	https://github.com/ramda/ramda
Underscore.js	http://underscorejs.org

grammiersprache, unterstützt aber verschiedene funktionale Prinzipien und Konzepte. Möchte man die funktionalen Konzepte nicht selbst implementieren, kann man auf eine der zahlreichen zur Verfügung stehenden funktionalen Bibliotheken zurückgreifen, die in **Tabelle 2** aufgelistet sind.

Im nächsten Beitrag dieser Artikelserie geht es um die reaktive Programmierung, die der funktionalen Programmierung nicht ganz unähnlich ist. ■



Philip Ackermann

arbeitet beim Fraunhofer-Institut für Angewandte Informationstechnologie FIT in den Bereichen Web Compliance, IoT und eHealth. Er ist Autor mehrerer Fachbücher und Fachartikel über Java und JavaScript.

<http://philipackermann.de>



.NET Developer Conference 2016

- **05.12.2016 – DevSessions:**
2 halbtägige Workshops
- **06.12.2016 – Konferenz:**
1 Keynote, 2 Themenstränge,
12 Vorträge
- **07.12.2016 – Workshops**
Ganztägiges Praxistraining

Köln, Pullman Cologne

Trends, Lösungen und Know-how für Profi-Entwickler

Themenauswahl:

- **Softwarequalität:**
Unit Testing, DDD, Refactor
- **Core:** .NET Core, Strings,
Async/Await
- **Frontend:** UI Engineering,
WPF, ASP.NET Core
- **Any App:** UWP, Xamarin,
Microservices



dotnet-developer-conference.de

#netdc16



DDConference

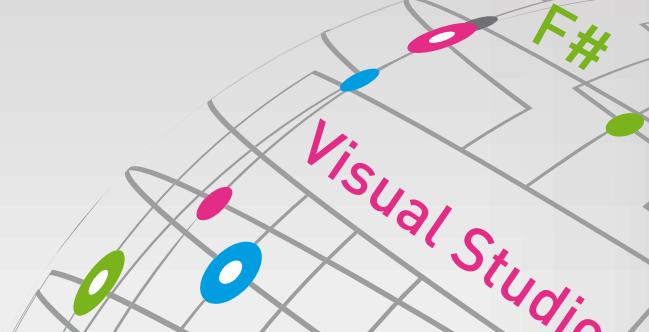
Veranstalter:  **developer
media**

Neue
Mediengesellschaft
Ulm mbH

Präsentiert von:  **dotnetpro**



.NET Developer Conference 2016



DevSessions: 05. Dezember 2016

	Frontend 1	Frontend 2	Softwarequalität 3	Architektur 4
09.00 – 13.00*	MVVM-Pattern mit WPF Bernd Marquardt Level 200	UWP Apps André Krämer Level 100	Software verbessern - mit aim42 Gernot Starke Level 100	Mikroservice Architektur mit Azure Service Bus N.N. Level 200
13.00 – 14.00	Mittagspause			
	Frontend 5	Frontend 6	Softwarequalität 7	Backend 8
14.00 – 18.00*	The WPF First Aid Kit David Würfel Level 300	Dreamteam: ASP.NET Core und Angular 2.0 Johannes Hoppe Level 100	TDD für Testmuffel Hendrik Lösch Level 200	Plattformunabhängiger Datenzugriff mit Entity Framework Core 1.0 Dr. Holger Schwichtenberg Level 100
*10.30 – 11.00 und 15.30 – 16.00 Kaffeepause				

Konferenz-Programm: 06. Dezember 2016



08.45	Begrüßung durch den Veranstalter		Begrüßung
09.00 – 09.55	Keynote: Welcome HoloLens • Damir Dobric und Andreas Erben		Keynote: Eine Branche im Goldrausch Constantin Klein
	Softwarequalität	Frontend	SMART DATA
10.00 – 10.55	Fachliche Architektur mit DDD für .NET Henning Schwentner Level 200	Windows 10 UI Engineering für WPF-Entwickler Thomas Immich Level 300	Datenqualität: Big Data Matching Dr. Hanna Köpcke
11.00 – 11.30	Kaffeepause		Kaffeepause
11.30 – 12.25	12 tips for unit tests that don't cripple your codebase Dennis Doomen	Datenbindung Deluxe – Deep Dive in das Binding von WPF Christian Giesswein Level 400	4x4: Vier Real-World Beispiele bei vier Cloud Providern Danny Linden
12.30 – 13.25	Refactoring C# Legacy Code Stefan Lieser Level 300	Running your ASP.NET Core app in a Docker container Maurice de Beijer	NoSQL: Einführung in Graphdatenbanken mit Neo4j Tobias Trelle
13.30 – 14.30	Mittagspause		Mittagspause
14.00 – 14.30	Lunchsession		"German Cloud" für Dev und IT-Pro Dobric und Erben
	Core	Any App	SMART DATA
14.30 – 15.25	.NET Core David Tielke Level 200	UWP - New Horizons Marek Pohanka und Daniel Bauer Level 200	Recommender-Algorithmen mit R, Spark DataFrames und Spark MLlib Henrik Behrens
15.30 – 16.00	Kaffeepause		Kaffeepause
16.00 – 16.55	Async/Await: Out of Context Tim Bussmann Level 200	Xamarin - .NET für die Hosentasche Sven-Michael Stübe Level 200	Smart Analytics: Streaming mit Apache Flink Stephan Papp
17.00 – 18.00	Die dunkle Wahrheit über Strings im .NET Bereich Christian Giesswein Level 400	Microservices and Azure Service Fabric Johannes C. Dumitru Level 300	Visual, Active & Mobile Reports nach HICHERT@SUCCESS mit IBM Cognos Holger Gerhards
18.30 – 20.30	Abendveranstaltung mit Night Coding		

Ihre Referenten (u.a.)



Damir Dobric,
DAENET Corporation



Dennis Doomen,
.NET Architect,
Coach and Writer



Christian Giesswein,
Giesswein Software-Solutions



Torsten Helmich,
SW Architekt und
Scrum Master



Johannes Hoppe,
HAUS HOPPE ITS



Stefan Lieser,
Mitgründer der
Clean Code Developer Initiative



Gernot Starke,
Gründungsmitglied
des iSAQB e.V.



David Tielke,
david-tielke.de



Ralf Westphal,
Mitgründer der
Clean Code Developer Initiative

Training: 08. Dezember 2016

developer media Training

Softwarequalität

David Tielke

Uhrzeit: 09.00 – 18.00 Uhr

Als erfahrener Trainer zeigt Ihnen David Tielke, wie Sie Softwarequalität spielend einfach in Ihren eigenen Projekten unterbringen, was die ersten Schritte sind und wie Schritt-für-Schritt ein mächtiger, nachhaltiger und transparenter Prozess für alle Beteiligten aufgebaut wird. Dabei liegt der Fokus auf allen Arten der Softwarequalität: richtig guten Code schreiben, kontrollieren, Architektur, Toolunterstützung, Testing ...

10% Rabatt bei Kombi-Anmeldung mit DDC

Workshops: 07. Dezember 2016

Workshop 1

Moderne Anwendungsentwicklung mit .NET

David Tielke

Uhrzeit: 09.00 – 18.00 Uhr

In diesem Workshop verschafft Ihnen Trainer David Tielke einen Überblick über die verschiedensten Anwendungsarten: moderne Desktopanwendung mit WPF, Datenzugriff mit dem Entity Framework, moderne Webanwendungen mit ASP.NET MVC und WebAPI, Windows 10 Universal Apps, Anwendung mit Windows IoT auf einem Raspberry PI, Anwendung in die Cloud bringen.

Workshop 2

Architektur Deluxe mit C# und .NET

Christian Giesswein

Uhrzeit: 09.00 – 18.00 Uhr

In diesem Workshop geht es darum, Merkmale von "gutem" Code zu erläutern und auch die Techniken. Zum Beispiel lernen Sie Dependency Injection und Inversion-Of-Control praxisnah für jedes Projekt kennen, damit auch Ihre Software innerhalb kürzester Zeit eine höhere Qualität aufweist.

Workshop 3

TDD 2.0 – Refactoring unnötig, offensichtlich, unvermeidbar

Ralf Westphal

Uhrzeit: 09.00 – 18.00 Uhr

Lassen Sie sich überraschen, wie "grüner Code" einerseits schon refaktoriert entstehen kann – oder andererseits die Refaktorisierung unvermeidbar wird. Einführung in "TDD as if you meant it" und "Informed TDD".

Workshop 4

Angular 2: Single-Page-Anwendungen entwickeln

Johannes Hoppe, Gregor Woiwode

Uhrzeit: 09.00 – 18.00 Uhr

Das neue Framework setzt auf die Programmiersprache TypeScript, komponentenbasierte Entwicklung, eine neue Template-Syntax und ein verbessertes Tooling. Unter Anleitung lernen Sie das alles, das Setup einer Angular 2 App, Pipes und Direktiven sowie die Navigation mit dem neuen Komponenten Router, Integration von Polymer & Kondo UI, Tooling, Unit-Test und Oberflächentests kennen. Danach beherrschen Sie Grundlagen und fortgeschrittene Themen.

Kooperationspartner (Stand 21.07.2016):





Foto: Shutterstock / filborg

DAS HISTORY API

Einblick in den Verlauf

Das History API ermöglicht den Zugriff auf den Browserverlauf per JavaScript.

Jedes Mal, wenn der Browser eine neue Webseite lädt oder auch, wenn innerhalb einer Webseite zu einer Sprungmarke (einem Anker) gesprungen wird, erzeugt er standardmäßig einen neuen Eintrag im Browserverlauf.

Über die Eigenschaft `history` des `window`-Objekts gelangt man an ein Objekt vom Typ `History`, das diesen Browserverlauf repräsentiert und über das zum einen Einblick in den Verlauf genommen werden kann, zum anderen aber auch der Verlauf geändert werden kann.

So repräsentiert die Eigenschaft `length` beispielsweise die Anzahl der Einträge im Browserverlauf, die Methode `back()` navigiert einen Eintrag zurück, die Methode `forward()` einen Eintrag weiter und die Methode `go()` zu einem bestimmten Eintrag (Tabelle 1):

```
window.history.length; // Anzahl der Einträge
window.history.back(); // zurück in der Historie
window.history.forward(); // weiter in der Historie
window.history.go(-2); // zwei Einträge zurück
window.history.go(2); // zwei Einträge weiter
window.history.go(0); // aktuelle Webseite erneut laden
```

Diese Eigenschaft und Methoden sind schon lange Teil des `history`-Objekts.

Zustände bei Single Page Applications

Mit dem Erfolg von Single Page Applications, in denen bekanntermaßen nur Teile einer Webseite beziehungsweise Anwendung neu geladen werden, musste jedoch eine neue

beziehungsweise zusätzliche Funktionalität her, um einzelne Einträge im Browserverlauf beschreiben zu können.

Der Grund für diese Funktionserweiterung: Werden nur Teile einer Webseite neu geladen, ändert sich standardmäßig

Tabelle 1: Eigenschaften und Methoden von History

Eigenschaft/ Methode	Beschreibung
<code>length</code>	Enthält die Anzahl an Einträgen in der Historie inklusive der aktuell geladenen Webseite.
<code>back()</code>	Geht zu der vorherigen Webseite in der Historie. Bewirkt das Gleiche, wie wenn der Nutzer die Zurück-Schaltfläche des Browsers betätigt.
<code>forward()</code>	Geht zu der nachfolgenden Webseite in der Historie. Bewirkt das Gleiche, wie wenn der Nutzer die Weiter-Schaltfläche des Browsers betätigt.
<code>go()</code>	Geht zu einer bestimmten Webseite in der Historie. Übergeben wird der Methode die Schrittweite ausgehend von der aktuellen Webseite. Beispielsweise sorgt der Wert <code>-1</code> dafür, dass zu der vorherigen Webseite gegangen wird, und der Wert <code>1</code> , dass zu der nachfolgenden Webseite gegangen wird. Wird ein Wert übergeben, für den es keine entsprechende Webseite in der Historie gibt, macht diese Methode nichts. Wird die Methode dagegen ohne Wert oder mit dem Wert <code>0</code> aufgerufen, wird die aktuelle Webseite erneut geladen.

dadurch nicht der URL, wodurch es folglich auch keinen neuen Eintrag im Browserverlauf gibt.

Zu Hilfe nahm man daher lange Zeit den Fragment Identifier des URL (sprich, was hinter dem #-Zeichen kommt). Werden Teile einer Webseite neu geladen, ändert man dabei einfach den Fragment Identifier und sorgt somit dafür, dass ein neuer Eintrag im Browserverlauf erzeugt wird. Wegen des enthaltenen #-Zeichens werden diese Art von URLs bekanntermaßen auch Hash-Bang-URLs genannt:

```
<body>
  <nav>
    <ul>
      <li><a href="#home">Startseite</a></li>
      <li><a href="#services">Services</a></li>
      <li><a href="#skills">Skills</a></li>
      <li><a href="#aboutus">Über Uns</a></li>
      <li><a href="#contact">Kontakt</a></li>
    </ul>
  </nav>

  <main id="content"></main>
</body>
```

Diese Technik ist allerdings mehr schlecht als recht, hat sie doch einige Einschränkungen. So steht beispielsweise der Fragment Identifier nur auf Clientseite zur Verfügung und wird nicht an den Server mitgeschickt.

Die HTML5-Features des History API

Seit einiger Zeit gibt es daher im Rahmen des sogenannten History API einige Anpassungen an dem *history*-Objekt, die das Arbeiten mit dem Browserverlauf auch für oben genannte Fälle erleichtern und mittlerweile von allen namhaften Browsern unterstützt werden (Bild 1).

Das History API definiert eine neue Eigenschaft und zwei neue Methoden: Die Eigenschaft *state* enthält den aktuellen Eintrag im Browserverlauf, über die Methode *pushState()* lässt sich ein neuer Eintrag im Verlauf erzeugen, und über die Methode *replaceState()* lässt sich der aktuelle Eintrag ersetzen (Tabelle 2).

Tabelle 2: Die neueren Eigenschaften und Methoden

Eigenschaft/ Methode	Beschreibung
<i>state</i>	Enthält den aktuellen Eintrag im Browserverlauf.
<i>pushState()</i>	Fügt der Historie einen Eintrag hinzu. Übergeben werden ein beliebiges JavaScript-Objekt (das sogenannte Zustandsobjekt), der Titel und optional ein URL.
<i>replaceState()</i>	Ersetzt den aktuellen Eintrag in der Historie durch einen neuen Eintrag. Übergeben werden ein Zustandsobjekt, der Titel und optional ein URL.

Um nun mit Hilfe dieser neuen Features die Zustände einer Single Page Application zu verwalten, verzichtet man auf Hash-Bang-URLs und verwendet normale URLs, wie im HTML-Code im folgenden Listing zu sehen.

```
<body>
  <nav>
    <ul>
      <li><a href="/home">Startseite</a></li>
      <li><a href="/services">Services</a></li>
      <li><a href="/skills">Skills</a></li>
      <li><a href="/aboutus">Über uns</a></li>
      <li><a href="/contact">Kontakt</a></li>
    </ul>
  </nav>
  <main id="content"></main>
</body>
```

In Listing 1 sehen Sie den entsprechenden JavaScript-Code. Für jeden Link auf der Navigation wird ein Event Listener registriert, der die Methode *pushState()* aufruft und damit dem Browserverlauf einen neuen Eintrag hinzufügt.

Als erstes Argument wird hierbei der anzuzeigende Inhalt übergeben, welcher der Einfachheit halber in dem Objekt *contents* enthalten ist (in der Praxis würde der Inhalt eher dynamisch vom Server geladen).

Als zweites Argument wird der Titel übergeben, der in der Titelleiste des Browsers angezeigt werden soll, wenn der entsprechende Link angeklickt wurde. Dazu wird einfach der Textinhalt des jeweiligen Link-Elements verwendet (*event.target.textContent*).

Als drittes Argument wird schließlich der URL angegeben, der in der Adresszeile angezeigt werden soll. Hier wird im Beispiel der Inhalt des *href*-Attributs des entsprechenden Link-Elements verwendet (*event.target.href*).

Der Aufruf von *preventDefault()* auf dem Event-Objekt verhindert zudem, dass die Standardaktion bei Betätigung der Links ausgeführt wird. Ließe man diesen Aufruf weg, würde der Browser den URL des Links einfach neu laden. ►



Das History API wird von fast allen Browsern unterstützt (Bild 1)

Jedes Mal, wenn der aktuelle Eintrag im Browserverlauf wechselt (beispielsweise, wenn die Zurück-Schaltfläche im Browser betätigt wird), löst der Browser das *popstate*-Event aus. Momentan wird allerdings der Inhalt der Webseite dann nicht aktualisiert, weil die Methode *updateContent()* nur bei Klick auf einen der Links aufgerufen wird, nicht aber bei Verwenden der Zurück-Schaltfläche (und im Übrigen auch nicht beim Verwenden der Weiter-Schaltfläche).

Auf Änderungen im Browserverlauf reagieren

Um das zu erreichen und damit den Inhalt des *<main>*-Elements auch beim Verwenden der Browser-Schaltflächen zu aktualisieren, muss man einfach die Methode *updateContent()* innerhalb eines Event Listeners für das *popstate*-Event aufrufen.

Dem Event Listener wird dabei ein Event-Objekt vom Typ *PopStateEvent* übergeben, über dessen Eigenschaft *state* man Zugriff auf den jeweils aus dem Browserverlauf entfernten Eintrag hat. Im folgenden Beispiel wird dessen Eigenschaft *content* ausgelesen und dem Container-Element für den Inhalt hinzugefügt:

```
window.addEventListener('popstate', (event) => {
  updateContent(event.state.content);
});
```

Um den aktuellen Eintrag im Browserverlauf durch einen anderen Eintrag zu ersetzen, kann man die Methode *replace-*

State() verwenden. Wie auch der Methode *pushState()* werden dieser Methode das Zustandsobjekt und optional Name sowie URL übergeben.

Fazit

Über das History API lässt sich auf den Browserverlauf zugreifen. Allerdings ist das Verhalten der Browser bezüglich des API leider nicht konsistent, beispielsweise wenn es um den Zeitpunkt geht, zu dem das *popstate*-Event ausgelöst wird.

Eine interessante Alternative zur Nutzung des History API ist daher die Bibliothek *history.js* (<https://github.com/browsersstate/history.js>), deren API mehr oder weniger gleich dem History API ist, dabei aber Browserunregelmäßigkeiten glättet. Außerdem nett: Für ältere Browser, die die neuen Methoden des History API überhaupt nicht unterstützen, wird auf Hash-Bang-URLs ausgewichen. ■



Philip Ackermann

entwickelt seit 15 Jahren Web- und Software-Anwendungen. Er arbeitet beim Fraunhofer-Institut für Angewandte Informationstechnologie FIT in den Bereichen Web Compliance, IoT und eHealth. Sie erreichen ihn über:

<http://philipackermann.de>

Listing 1: Hinzufügen von Einträgen in die Browserhistorie

```
function init() {
  let contentElement = document.getElementById(
    'content');
  let contents = {
    home: {
      content: 'Startseite'
    },
    services: {
      content: 'Dienste'
    },
    skills: {
      content: 'Skills'
    },
    aboutus: {
      content: 'Über Uns'
    },
    contact: {
      content: 'Kontakt'
    }
  };
  function handleClick(event) {
    let pageName =
      event.target.getAttribute('href').split('/').pop();
    let content = contents[pageName];

    updateContent(content.content);
    history.pushState(
      content,
      event.target.textContent,
      event.target.href
    );
    return event.preventDefault();
  }
  // Registrieren der Event Listener
  let linkElements =
    document.getElementsByTagName('a');
  for (let i = 0; i < linkElements.length; i++) {
    linkElements[i].addEventListener(
      'click',
      handleClick,
      true
    );
  }
  function updateContent(content) {
    contentElement.textContent = content;
  }
  document.addEventListener('DOMContentLoaded', init);
}
```


Updates für Ihr Know-how

„Erfolgreiche Entwickler haben den für ihre Software passenden Testmix. Lernen Sie, wie man externe wie interne Qualität effizient sicherstellt.“

Sebastian Bergmann
PHP-Consultant und Trainer



Apps für Windows 8/10 entwickeln

Trainer: Lars Heinrich

2 Tage, 08.-09.12.2016, Köln
Ab 1.799 EUR zzgl. MwSt.



Moderne Webentwicklung mit ASP.NET Core

Trainer: David Tielke

3 Tage, 21.-23.11.2016, Köln
Ab 2.199 EUR zzgl. MwSt.



Hybrid-Apps mit Ionic, Cordova und Angular

Trainer: Hendrik Lösch

2 Tage, 14.-15.11.2016, Köln
Ab 1.799 EUR zzgl. MwSt.



PHPUnit erfolgreich einsetzen

Trainer: Sebastian Bergmann

2 Tage, 07.-08.11.2016, Köln
Ab 1.799 EUR zzgl. MwSt.



Ihr Ansprechpartner:

Fernando Schneider – Key Account Manager – developer media

Telefon: +49 (0)89 74117-831 – E-Mail: fernando.schneider@developer-media.de

NAVIGATION MIT QT QUICK CONTROLS 2

Viele Varianten

Die neuen Qt Quick Controls 2 bieten viele Varianten der Navigation durch eine App an.

In der vorangegangenen Ausgabe der **web & mobile developer** (Seite 14) habe ich bereits eine Einführung in die neuen leichtgewichtigen Qt Quick Controls 2 gegeben, die Projektstruktur erläutert sowie die Verwendung der Farben, Fonts und Icons in Google Material Style beschrieben. Wie einfach Controls customized werden können, wurde an den Floating Action Buttons demonstriert.

Eine mobile App besteht nicht nur aus Controls – es muss auch einfach sein, durch die verschiedenen Seiten navigieren zu können.

Qt Quick Controls 2 – Navigation Controls

Qt Quick Controls 2 lassen keine Wünsche offen, wenn es darum geht, dem Anwender einen einfachen Weg der Navigation anzubieten: Page, StackView, SwipeView, TabBar, Bottom Navigation, Drawer, ToolBar und Menu.

Im vorigen Heft habe ich eine einfache App mit nur einer langen Seite mit Scrollbar genutzt. Es gibt eine Menge Anwendungen, wo das ausreicht, aber meist sind es doch mehrere Seiten, durch die der Anwender navigieren möchte.

Bei einer StackView werden Seiten in einem Stapel aufeinander gelegt, eine SwipeView vereint alle Seiten nebeneinander und nur der sichtbare Ausschnitt wird verschoben. TabBar ist geeignet, wenn wir zwei bis drei Bereiche zur Auswahl anbieten. Die Bottom Navigation ist für drei bis fünf Anwendungsbereiche konzipiert, und eine Drawer Navigation bietet sich an, wenn es mehr als fünf darstellbare Ziele gibt.

Menus und ToolBars sowie Floating Action Buttons sind weitere Elemente, die zur Navigation verwendet werden.

Beim Einsatz eines Smartphones mit Tastatur (beispielsweise BlackBerry PRIV mit Android M) oder einer angeschlossenen Bluetooth-Tastatur sind Shortcuts sinnvoll, um schnelle Wege anzubieten.

StackView

Beim Einsatz einer StackView werden Seiten aufeinander gestapelt: Mit *push()* kommt eine Seite obendrauf, mit *pop()* wird die oberste entfernt. Qt ermöglicht auch, ein Array an Seiten zu pushen oder zu entfernen, Seiten im Stapel zu suchen oder direkt zu einer bestimmten Position zu gehen.

Um einen Stapelvorgang auszulösen, muss der User natürlich eine entsprechende Aktion triggern wie beispielsweise »Zeige Position zu Auftrag« oder »Zeige Adresse des Kunden«.

Üblicherweise kommen dazu Buttons zum Einsatz. Im vorigen Heft haben wir bereits den Floating Action Button (FAB) kennengelernt. Normale Buttons können im Material Style

flach oder erhoben sein. Qt Quick Controls 2 bietet einen Standard-Button an, der leicht zu customize ist. In der StackView-Beispiel-App nutze ich beide Varianten, um den Unterschied aufzuzeigen. **Listing 1** zeigt die customized Buttons, **Listing 2** den Aufruf. **Bild 1** zeigt Flat und Raised Buttons aus der Beispiel-App.

Eine StackView wird im ApplicationWindow wie folgt definiert:

```
StackView {
    id: navPane
    focus: true
    anchors.fill: parent
    initialItem: pageOne
    // ....
} // navPane
Component {
    id: pageOne
    PageOne {
    }
} // pageTwo
Component {
    id: pageTwo
    PageTwo {
```



Foto: Shutterstock / Vectori

```

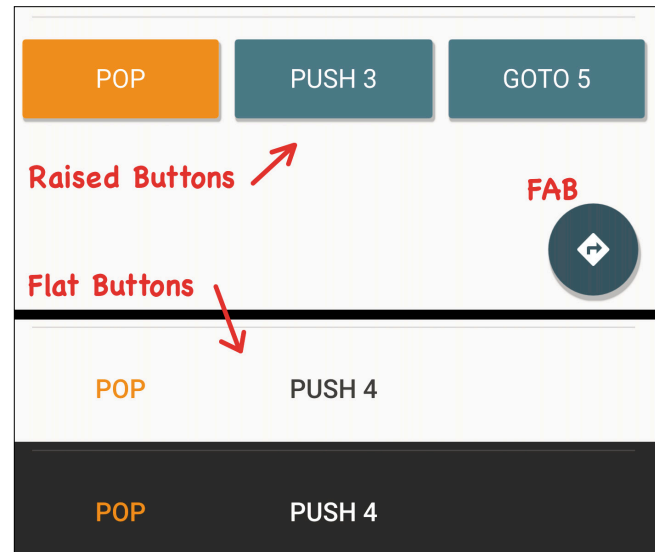
}
} // pageTwo
// ...

```

Die StackView füllt das ApplicationWindow immer komplett aus. Damit beim Starten der App auch sofort etwas sichtbar ist, sollte das Root-Element als *initialItem* definiert sein.

Mögliche Seiten, die auf den Stapel gepackt werden sollen, werden am einfachsten als Component definiert, denn dann übernimmt die StackView automatisch das Instanzieren und Laden einer Komponente beim *push()* sowie das Zerstören beim *pop()*. Werden mehrere Seiten als Array per *push()* auf den Stapel gelegt, dann kann sogar das genaue Verhalten vorgegeben werden, ob alle Seiten instanziiert werden oder erst dann, wenn sie sichtbar werden.

Ich bin ein großer Fan von Lazy Loading und dynamischen Komponenten, um Anwendungen performant zu halten und nur notwendige Objekte im Speicher zu haben. Bei einer ►



Flat und Raised Buttons aus der Beispiel-App (Bild 1)

Listing 1: Customized Buttons

```

Button {
    id: button
    property alias textColor: buttonText.color
    property alias buttonColor: buttonBackground.color
    focusPolicy: Qt.NoFocus
    Layout.fillWidth: true
    Layout.preferredWidth : 1
    leftPadding: 6
    rightPadding: 6
    contentItem: Text {
        id: buttonText
        text: button.text
        opacity: enabled ? 1.0 : 0.3
        color: textOnPrimary
        horizontalAlignment: Text.AlignHCenter
        verticalAlignment: Text.AlignVCenter
        elide: Text.ElideRight
        font.capitalization: Font.AllUppercase
    }
    background:
        Rectangle {
            id: buttonBackground
            implicitHeight: 48
            color: primaryColor
            radius: 2
            opacity: button.pressed ? 0.75 : 1.0
            layer.enabled: true
            layer.effect: DropShadow {
                verticalOffset: 2
                horizontalOffset: 1
                color: dropShadow
                samples: button.pressed ? 20 : 10
                spread: 0.5
            }
        }
    } // background
} // button

// ButtonFlat.qml
Button {
    id: button
    property alias textColor: buttonText.color
    focusPolicy: Qt.NoFocus
    Layout.fillWidth: true
    Layout.preferredWidth : 1
    leftPadding: 6
    rightPadding: 6
    contentItem: Text {
        id: buttonText
        text: button.text
        opacity: enabled ? 1.0 : 0.3
        color: flatButtonTextColor
        horizontalAlignment: Text.AlignHCenter
        verticalAlignment: Text.AlignVCenter
        elide: Text.ElideRight
        font.capitalization: Font.AllUppercase
        font.weight: Font.Medium
    }
    background:
        Rectangle {
            id: buttonBackground
            implicitHeight: 48
            Layout.minimumWidth: 88
            color: button.pressed ?
                buttonText.color : "transparent"
            radius: 2
            opacity: button.pressed ? 0.12 : 1.0
        }
    } // background
} // button

```

StackView geschieht das mit Komponenten automatisch. Ich kann aber auch Items direkt definieren oder als Loader gestalten, die dann das enthaltene Objekt instanzieren.

Ich kann die StackView jederzeit fragen, wie viele Seiten auf dem Stapel liegen (*depth*), was sinnvoll ist, wenn es um die Android-Standardnavigation geht:

- Zurück mit dem Button in der Titelzeile oben links,
- Zurück mit der Android-System-Back-Taste.

Hier entscheide ich, ob der Button in der Titelzeile sichtbar wird:

```

ToggleButton {
    enabled: navPane.depth > 1
    focusPolicy: Qt.NoFocus
    Image {
        id: backImageImage
        visible: navPane.depth > 1
        anchors.centerIn: parent
        source: "qrc:/ges/" +
            iconOnPrimaryFolder+ "/arrow_back.png"
    }
    onClicked: {
        navPane.popOnePage()
    }
}

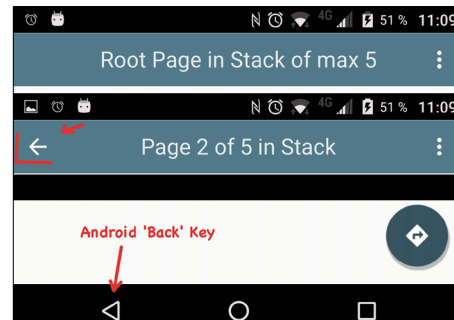
```

Und hier das Verhalten der Android-Back-Taste:

```

StackView {
    id: navPane
    focus: true
    anchors.fill: parent
    initialItem: pageOne
    // support of BACK key
    Keys.onBackPressed: {
        event.accepted = navPane.depth > 1
        popOnePage()
        if(navPane.depth == 1) {

```

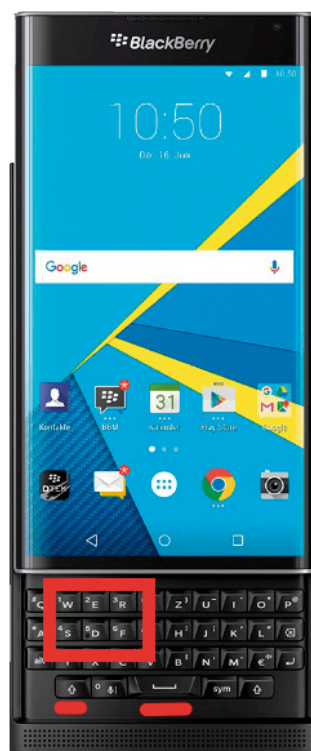


Zurück-Navigation bei Android
(Bild 2)

```

var page = navPane.get(0)
    page.cleanup()
}
// ...

```



PRIV Shortcuts: Schnell über Shortcuts navigieren (Bild 3)

Bild 2 zeigt beide Zurück-Buttons aus der Beispiel-App. Beim Einsatz von Smartphones mit Tastatur kann ich einfach und schnell über Shortcuts navigieren (Bild 3):

```

StackView {
    Shortcut {
        sequence: "s"
        onActivated: navPane.goToPage(4)
    }
    Shortcut {
        sequence: "Alt+s"
        onActivated: navPane.goToPage(4)
    }
    Shortcut {
        sequence: " "
        onActivated: navPane.pushNextPage()
    }
}

```

Diese Shortcuts funktionieren auch über eine angeschlossene Bluetooth-Tastatur. In der bei GitHub verfügbaren Beispiel-App habe ich

Listing 2: Aufruf

```

RowLayout {
    // implicite fillWidth = true
    spacing: 10
    ButtonRaised {
        text: "Pop"
        buttonColor: accentColor
        onClicked: {
            navPane.popOnePage()
        }
    }
    ButtonRaised {
        text: "Push 3"
        onClicked: {
            navPane.pushOnePage(pageThree)
        }
    }
    ButtonRaised {
        text: "GoTo 5"
        onClicked: {
            navPane.goToPage(5)
        }
    }
} // button row

```


verschiedene Varianten der Navigation durch einen Stack eingebaut. **Bild 4** zeigt, was unter der Haube passiert, wenn man auf die entsprechenden Buttons klickt.

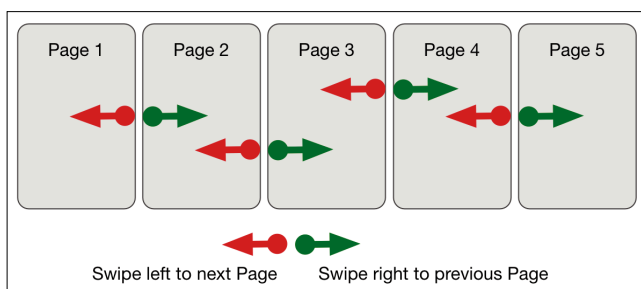
Die StackView ist ein mächtiges Instrument, das sich immer dann anbietet, wenn der Anwender selbst entscheidet, wann er was sehen möchte. Somit kann der Stapel ganz unterschiedlich aussehen – je nachdem, auf welche Aktionsbuttons der User geklickt hat.

SwipeView

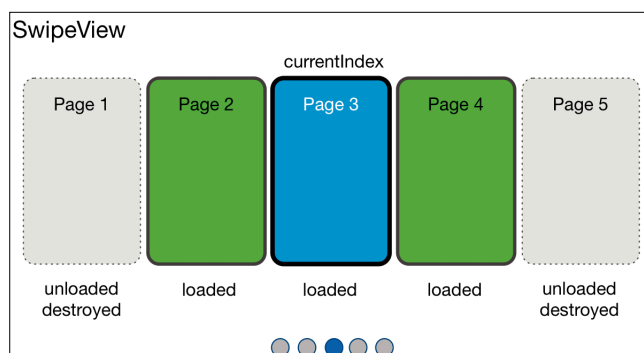
Bild 5 zeigt die Navigation durch mehrere Seiten in einer SwipeView. Wie man sieht, ist die Reihenfolge vorgegeben: Alle Seiten liegen nebeneinander. Es können dabei auch einzelne Seiten ein- und ausgeblendet werden, aber der Anwender hat in der Regel keinen Einfluss auf die Anordnung und Reihenfolge.

SwipeView eignet sich für einen Wizard, in dem der Anwender nacheinander Fragen beantwortet oder Daten eingibt. Eine SwipeView kann auch zusammen mit einer TabBar genutzt werden.

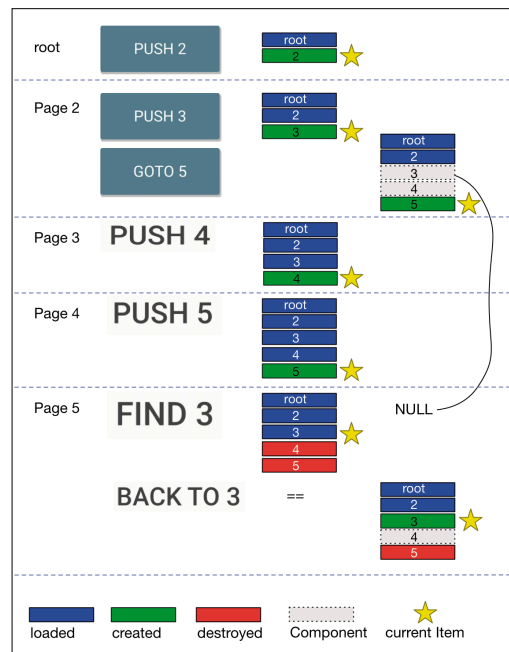
Leider ist bei der SwipeView nicht eingebaut, dass die Seiten als Komponenten definiert werden und erst bei Bedarf instanziiert werden. Ich habe aber die Info bekommen, dass es



SwipeView mit Gesten: Navigation über mehrere Seiten (**Bild 5**)



SwipeView dynamisches Laden: Diese Seiten sind geladen (**Bild 6**)



StackView-Workflow: Ein Blick unter die Haube (**Bild 4**)

hierfür in Qt 5.8 mit den QtQuick-Controls 2.1 Unterstützung geben wird – bis dahin habe ich das als Beispiel selbst implementiert.

Während des Wischens nach rechts oder links sind ja immer die folgende und die vorige Seite sowie die aktuelle Seite sichtbar. Daher ist es sinnvoll zu sagen, dass immer die aktuelle, vorige und nächste Seite geladen sind und alle, die herausgeschoben werden, entladen werden.

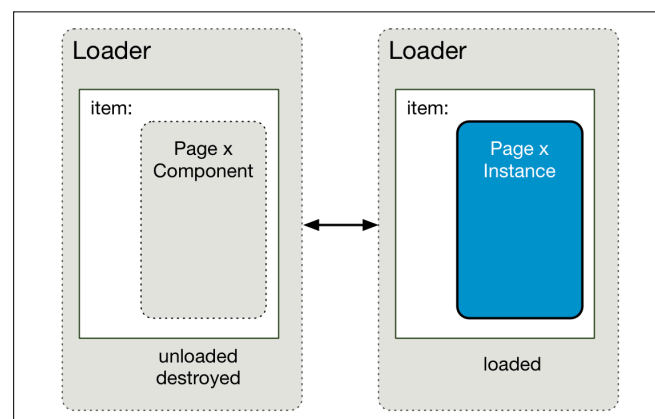
Bild 6 demonstriert, welche Seiten jeweils geladen sind. Eine andere Variante könnte die Seiten schrittweise während der Bearbeitung laden und dann geladen lassen.

Damit der Benutzer weiß, auf welcher Seite er sich befindet und dass es Folgeseiten gibt, bietet es sich an, einen PageIndicator zusammen mit der SwipeView anzu-

zeigen. Der PageIndicator und die SwipeView sind durch die `currentIndex`-Property verbunden:

```
SwipeView {
    id: navPane
    // ...
    currentIndex: 0
    // currentIndex is the NEXT index swiped to
    onCurrentIndexChanged: {
        // ...
    }
}

PageIndicator {
    id: pageIndicator
    count: navPane.count
    currentIndex: navPane.currentIndex
```



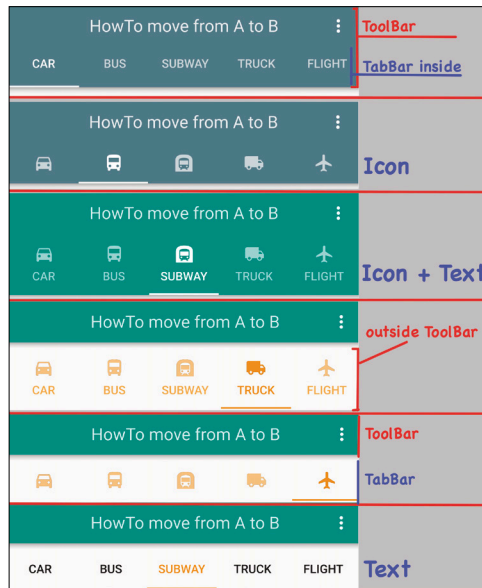
Loader Button: Seiten dynamisch laden und wieder entladen (**Bild 7**)

```
anchors.bottom: navPane.bottom
anchors.horizontalCenter:
parent.horizontalCenter
}
```

Um die Seiten dynamisch laden und wieder entladen zu können, nutzen wir einen Loader (Bild 7). Ein Loader ist ein Platzhalter für ein UI Control, der erst geladen wird, wenn der Loader auf *active* gesetzt wird. Auf das eigentliche Control lässt sich dann per *.item*-Property zugreifen.

Eine beispielhafte Loader-Definition für die Seite mit dem *currentIndex* 2 der *SwipeView* sieht so aus:

```
Loader
{
    id: pageThreeLoader
    property string title: active?
    item.title: "... "
    active: navPane.currentIndex == 1 ||
```



TabBar-Designvarianten: Diese TabBars lassen sich mit Qt Quick Controls 2 erstellen (Bild 8)

```
navPane.currentIndex == 2 ||
navPane.currentIndex == 3
source: "pages/PageThree.qml"
onLoaded: item.init()
}
```

Wir sehen hier wieder, wie einfach Signale zwischen verschiedenen UI Controls bei Qt miteinander verbunden werden. Das *active*-Property des Loaders wird auf *true* gesetzt, wenn der *currentIndex* der *SwipeView* 1, 2 oder 3 ist.

Sobald die Seite geladen wurde, wird noch eine *init()*-Funktion ausgeführt. Wird die Seite dann weiter weggeschoben, ändert sich *active* automatisch in *false* und die Seite wird entladen (zerstört).

Wer das Beispiel aus GitHub ausprobiert, wird sehen, dass – egal wie schnell man durch die *SwipeView*

wischt – die Seiten ohne Ruckeln geladen werden.

Listing 3: TabButton

```
TabButton {
    property color theButtonColor: accentColor
    property string theIconFolder: iconFolder
    property alias hasOverlay: colorOverlay.visible
    property real theOpacity: 1.0
    focusPolicy: Qt.NoFocus
    height: 48
    contentItem:
        Item {
            Image {
                id: contentImage
                anchors.centerIn: parent
                horizontalAlignment: Image.AlignHCenter
                verticalAlignment: Image.AlignVCenter
                source: "qrc:/images/"+theIconFolder+"/"+
                    modelData.icon
                opacity: colorOverlay.visible? 1.0 :
                    theOpacity
            }
            ColorOverlay {
                id: colorOverlay
                visible: true
                anchors.fill: contentImage
                source: contentImage
                color: index == navPane.currentIndex ?
                    theButtonColor : Qt.lighter(theButtonColor)
            }
        } // item
}
```

TabBar-Varianten

Der Google Material Design Guide erlaubt unterschiedlichste Varianten für eine Navigation mit Tabs:

- Tabs mit Text,
- Tabs mit Icons,
- Tabs mit Text und Icons.

Tabs können ohne eine Titelzeile genutzt werden, innerhalb einer Titelzeile, oder sie können auch erst im Content-Bereich definiert werden, wenn beispielsweise über den Tabs noch Informationen angezeigt werden sollen. All diese Variationen können mit den Qt Quick Controls 2 gebaut werden (Bild 8). Listing 3 zeigt einen TabButton mit einem Icon.

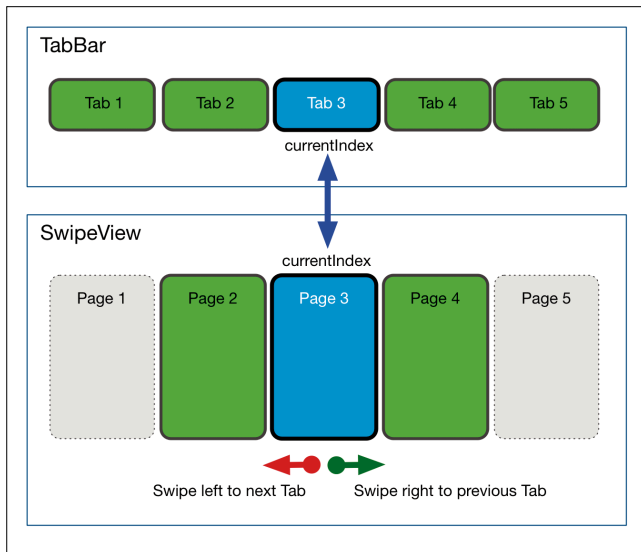
Eine TabBar hat ebenso wie eine *SwipeView* einen *currentIndex*, und da es gemäß Google Material Styleguide möglich sein muss, durch Tabs auch per Wischgeste zu navigieren, bietet sich die Kombination von TabBar und *SwipeView* an (Bild 9).

Datenmodell für einzelne Tabs

In der TabBar-Beispielanwendung zeige ich einen weiteren Weg auf, wie UI Controls angelegt werden können, die gleiche Strukturen aufweisen. Dazu definiere ich ein Datenmodell für die einzelnen Tabs:

```
property var tabButtonModel:
[{"name": "Car", "icon": "car.png"},
{"name": "Bus", "icon": "bus.png"},
{"name": "Subway", "icon": "subway.png"},
{"name": "Truck", "icon": "truck.png"},
{"name": "Flight", "icon": "flight.png"}]
```

Jeder Tab hat einen Namen und ein zugehöriges Icon.



TabBar-Navigation: Kombination von TabBar und SwipeView (Bild 9)

Mit Hilfe eines Repeaters erzeuge ich jetzt die einzelnen TabButtons:

```
Repeater {
    model: tabButtonModel
    TabButton {
        text: modelData.name
        width: ...
    }
} // repeater
```

Auf die Elemente des Models kann über *modelData* zugegriffen werden. Das Arbeiten mit den Tabs kann im Detail in der Beispielanwendung bei GitHub nachvollzogen werden.

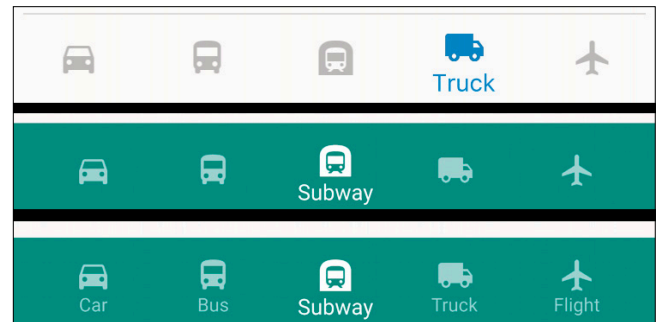
Bottom Navigation

Unter Android gab es lange etwas, das ich vermisst habe: die Navigation aus einer Toolbar, die sich unten befindet (Bottom) – so wie es bei iOS und BlackBerry 10 möglich ist. Google hat das aber auch erkannt, und so gibt es jetzt als Bestandteil des Material Design Guides auch die Bottom Navigation.

Bottom Navigation soll für mindestens drei und maximal fünf Auswahlbereiche eingesetzt werden.

Die Bottom Navigation Bar kann mit der Primärfarbe hinterlegt sein, oder farblos und nur das ausgewählte Icon ist farbig markiert. Bei Android wird – wenn es mehr als drei Buttons sind – Text nur beim aktuell ausgewählten angezeigt – iOS zeigt normalerweise Text zu allen Buttons. In der Beispiel-App habe ich alle Varianten implementiert (Bild 10).

Die Bottom Navigation ergibt keinen Sinn, wenn das Gerät in den Landscape-Modus gedreht wird, da sie dann zu viel in der Höhe wegnimmt und mit so wenigen Buttons über die gesamte Breite verteilt auch nicht gut aussieht. Im Landscape-Modus zeige ich daher die Buttons links untereinander an. Zur Erzeugung der Navigationsleiste nutze ich wieder ein Datenmodel und einen Repeater (Listing 4). Der Navigation-Button ist ein customized Button, da es keinen ToolButton



Bottom-Navigation-Varianten: In der Beispiel-App implementiert (Bild 10)

speziell für die Bottom Navigation gab (Listing 5). Die Navigation in der App selbst ist über eine StackView gelöst, auf der sich immer nur ein einziges Control befindet, das dann je nach gewähltem Programmbereich eine Seite, eine StackView, SwipeView oder TabBar sein kann. Den Austausch der sichtbaren Seite löst die Änderung des *navigationIndex* aus.

Drawer

Wenn nun die maximal fünf Programmbereiche nicht ausreichen? Für diese Zwecke gibt es den Drawer, der per Swipe-geste vom linken Rand geöffnet werden kann oder durch den Menu-Button oben links. Ein Drawer kann unterschiedlichste Elemente beinhalten (Bild 11):

- Einen Header mit Programmname, Icon, Background und weiteren Infos, ▶

Listing 4: Repeater

```
property var navigationModel:
    [{"name": "Car", "icon": "car.png", "source":
      "../pages/PageOne.qml"},
     {"name": "Bus", "icon": "bus.png", "source":
      "../pages/PageTwo.qml"},
     {"name": "Subway", "icon": "subway.png",
      "source": "../pages/PageThree.qml"},
     {"name": "Truck", "icon": "truck.png", "source":
      "../pages/PageFour.qml"},
     {"name": "Flight", "icon": "flight.png",
      "source": "../pages/PageFive.qml"}]
property int navigationIndex: 0
onNavigationIndexChanged: {
    rootPane.activeDestination(navigationIndex)
}

Repeater {
    model: navigationModel
    NavigationButton {
        id: myButton
        isColored: false
    }
} // repeater
```

Listing 5: NavigationButton

```

ToolButton {
    id: myButton
    property bool isActive: index == navigationIndex
    height: 56
    width: myBar.width / navigationModel.length
    Column {
        spacing: 0
        topPadding: myButton.isActive ||
            !suppressInactiveLabels? 0 : 6
        anchors.horizontalCenter: parent.horizontalCenter
        Item {
            anchors.horizontalCenter: parent.horizontalCenter
            width: 24
            height: 24
            Image {
                id: contentImage
                width: 24
                height: 24
                verticalAlignment: Image.AlignTop
                anchors.horizontalCenter: parent.horizontalCenter
                source: "qrc:/images/"+myIconFolder+"/"+
                    modelData.icon
                opacity: isActive? myBar.activeOpacity :
                    myBar.inactiveOpacity
            }
            ColorOverlay {
                id: colorOverlay
                visible: myButton.isColored && myButton.isActive
                anchors.fill: contentImage
                source: contentImage
                color: primaryColor
            }
        } // image and coloroverlay
        Label {
            visible: myButton.isActive ||
                !suppressInactiveLabels
            anchors.horizontalCenter: parent.horizontalCenter
            text: modelData.name
            opacity: isColored? (isActive? 1.0 : 0.7) :
                (isActive? myBar.activeOpacity :
                    myBar.inactiveOpacity)
            color: isColored? (isActive? primaryColor :
                flatButtonTextColor) : textOnPrimary
            font.pixelSize: myButton.isActive?
                fontSizeActiveNavigationButton :
                fontSizeInactiveNavigationButton
        } // label
    } // column
    onClicked: {
        navigationIndex = index
    }
} // myButton

```

- Navigation per Icon und Text,
- Navigation nur mit Text,
- Trennlinien (Divider),
- Zwischenüberschriften.

Zur Generierung der Drawer-Controls benutze ich wiederum ein Datenmodell, das etwas komplexer ist als die bisher verwendeten. Hier ist beispielhaft ein Element des NavigationModels:

```

[
    { "type": "../navigation/
      DrawerNavigationButton.qml",
      "name": "Home",
      "icon": "home.png",
      "source": "../navigation/
        HomeNavigation.qml",
      "showCounter": false,
      "showMarker": false,
      "a_p": 1},
    {...}
]

```

Durch die Angabe des *type* als QML-URL können ganz unterschiedliche Controls er-

zeugt werden (Divider, NavigationButtons et cetera). *showCounter* und *showMarker* ermöglichen genau das, was ihr Name vermuten lässt: einen Counter zu setzen oder die Farbe eines Markers.

Ein Drawer kann auf ganz unterschiedliche Programmbereiche verweisen, die entweder sehr selten genutzt werden (*About*, *Settings*) oder aber immer.

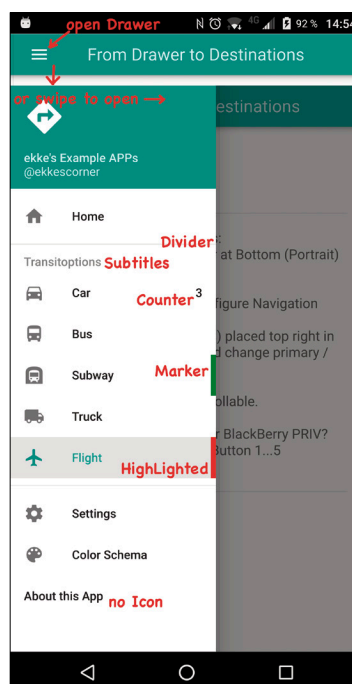
Hier ist es sinnvoll, mit einer Activation-Policy zu arbeiten. Ich habe in der Beispiel-App *IMMEDIATELY*, *LAZY* und *WHILE_CURRENT* implementiert:

```

Loader {
    id: pageLoader
    property int pageActivationPolicy:
    modelData.a_p
    active: pageActivationPolicy ==
        activationPolicy.IMMEDIATELY
    visible: false
}

```

Drawer-Design: Unterschiedlichste Elemente (Bild 11)



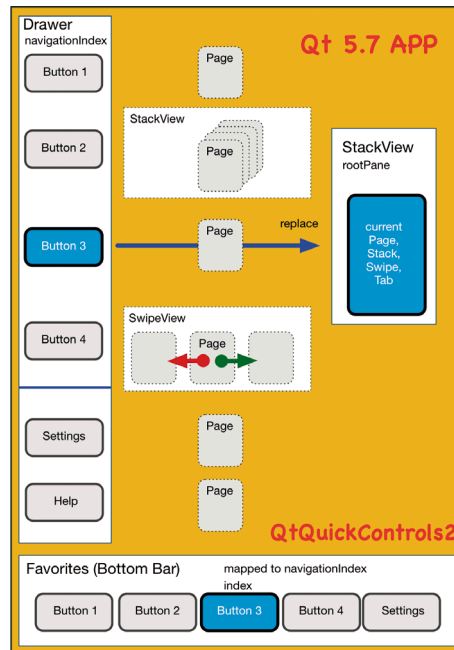

```

source: modelData.source
onLoaded: {
    item.init()
    if(pageActivationPolicy !=
activationPolicy.IMMEDIATELY)
    {
        rootPane.replaceDestination
        (pageLoader)
    }
}
}

```

IMMEDIATELY lädt Seiten sofort, *LAZY* beim ersten Zugriff und *WHILE_CURRENT* nur, solange sie genutzt werden. Wer schon mit BlackBerry 10 Cascades programmiert hat, wird erkennen, dass es bei der TabbedPane – die mit dem Drawer verglichen werden kann – etwas Ähnliches gab. Die Details der Implementierung können im Sourcecode nachgeschaut werden – das würde an dieser Stelle den Rahmen des Artikels sprengen.

Da mir die Bottom-Navigation sehr gut gefällt (kurze Fingerwege), habe ich diese mit dem Drawer kombiniert und



Komplex: Der Aufbau einer komplexen App (Bild 12)

zeige im Portrait-Modus alternativ zum Drawer die Navigationsleiste am Fuß an. Neben dem Menu-Button sind bis zu vier Favoriten zur schnellen Navigation sichtbar.

Dabei wird dann der Index der Bottom-Navigation-Bar auf den entsprechenden *NavigationIndex* des Drawers gemappt.

Navigation durch eine komplexe App

Spätestens jetzt mit den Navigation Controls sieht man sehr gut, dass die Qt Quick Controls 2 ausgezeichnet geeignet sind, um mobile Apps zu bauen. Der Vorteil liegt darin, mit nur einem UI-Code Apps unter Android, iOS und Windows 10 auszuführen. Bild 12 zeigt zusammenfassend den Aufbau einer komplexen App.

Alle Beispiele sind bei GitHub verfügbar – es ist also einfach auszuprobieren: Qt installieren, den QtCreator

öffnen und die Apps auf Android und iPhone ausführen.

Fazit

Qt 5.7 Quick Controls 2 bieten Navigation Controls an, mit denen sich einfache oder hochkomplexe Apps entwickeln lassen. Diese Navigation Controls sind einfach anzupassen oder zu erweitern – so ist es im Enterprise-Umfeld möglich, Apps zu bauen, die unter BlackBerry 10, Android, iOS und Windows einen ähnlichen Workflow anbieten. Wenn etwas fehlt, kann es leicht selbst ergänzt werden.

Es lohnt sich auch, Bugreports für weitergehende Wünsche zu erstellen. So vermisse ich generell in Android die Möglichkeit, auf eine darunterliegende Seite zu schauen, indem ich die obere Seite etwas nach rechts schiebe. Unter iOS ist so etwas bei den Einstellungen möglich, und BlackBerry hat das Feature peek-back genannt. Die gute Nachricht: Es wird auch in die Qt Quick Controls 2 eingebaut werden.

Im dritten und letzten Teil der Artikelserie sehen wir uns dann an, wie es mit Data Binding, Datenmodellen und Listen-Handling aussieht. Danach geht es dann weiter mit mobaDSL – dem Codegenerator für mobile Business Apps unter BlackBerry 10 und Qt 5.7 (Android, iOS, W10). ■

Links zum Thema

- Qt Quick Controls
<http://doc-snapshots.qt.io/qt5-5.7/qtquickcontrols2-index.html>
- Qt Quick Container
<http://doc-snapshots.qt.io/qt5-5.7/qtquickcontrols2-containers.html>
- Qt Quick 2 Navigation
<http://doc-snapshots.qt.io/qt5-5.7/qtquickcontrols2-navigation.html>
- Google Material Style
www.google.com/design/spec/material-design/introduction.html
- Qt Material Style
<http://doc-snapshots.qt.io/qt5-5.7/qtquickcontrols2-material.html>
- StackView-Beispiel-App
http://github.com/ekke/stacked_pages_x
- SwipeView-Beispiel-App
http://github.com/ekke/swiped_pages_x
- TabBar-Beispiel-App
http://github.com/ekke/tab_pages_x
- Bottom-Navigation-Beispiel-App
http://github.com/ekke/bottom_nav_x
- Drawer-Beispiel-App
http://github.com/ekke/drawer_nav_x



Ekkehard Gentz

ist Autor, Trainer und Speaker auf Konferenzen. Er entwickelt als Independent Software Architect mobile Anwendungen für internationale Kunden, ist BlackBerry Elite Member und bloggt unter:

<http://ekkes-corner.org>

IOS MAPKIT

Streckenplanung

Mit iOS MapKit lassen sich Karten problemlos in eigene Apps integrieren.

Das MapKit Framework von iOS ermöglicht es dem Entwickler, problemlos Karten in die eigene App zu integrieren. Neben den Karten können natürlich auch markante Punkte, kurz POIs (Points of Interest), innerhalb der Karte markiert werden. Hierzu kann man entweder das Standard-Icon oder ein selbst definiertes Icon verwenden.

Neben dem Icon lässt sich eine solche Markierung auch noch zusätzlich mit einem erläuternden Text versehen, der

bei einer Berührung des Displays an der entsprechenden Stelle angezeigt wird. Nützlich ist auch die Möglichkeit, mehrere solcher Punkte mittels der Klasse *MapItems* anlegen zu können.

Mit diesen Mitteln ist es beispielsweise möglich, einen kleinen Sightseeing-Tour für Anwender aufzubauen. Das ist schon ganz nett. Aber: Der Benutzer muss seinen Weg zwischen den entsprechenden Punkten immer noch selbst finden.

Listing 1: Kompletter Code der Klasse

```
import UIKit
import MapKit

class ViewController: UIViewController,
MKMapViewDelegate {
    @IBOutlet var mkMapView: MKMapView!

    override func viewDidLoad() {
        super.viewDidLoad()
        mkMapView.delegate = self

        let startPoint = CLLocationCoordinate2D(latitude:
52.525084, longitude: 13.369402)
        let endPoint = CLLocationCoordinate2D(latitude:
52.516275, longitude: 13.377704)
        let startMKPlacemark = MKPlacemark(coordinate:
startPoint, addressDictionary: nil)
        let endMKPlacemark = MKPlacemark(coordinate:
endPoint, addressDictionary: nil)
        let startMkMapItem = MKMapItem(placemark:
startMKPlacemark)
        let endMkMapItem = MKMapItem(placemark:
endMKPlacemark)
        let startAnnotation = MKPointAnnotation()
        startAnnotation.title = "Hauptbahnhof"
        if let location = startMKPlacemark.location {
            startAnnotation.coordinate = location.coordinate
        }
        let endAnnotation = MKPointAnnotation()
        endAnnotation.title = "Brandenburger Tor"
        if let location = endMKPlacemark.location {
            endAnnotation.coordinate = location.coordinate
        }

        self.mkMapView.showAnnotations([startAnnotation,
endAnnotation], animated: true )
        let mkDirectionsRequest = MKDirectionsRequest()
        mkDirectionsRequest.source = startMkMapItem

        mkDirectionsRequest.destination = endMkMapItem
        mkDirectionsRequest.transportType = .Automobile
        let mkDirections = MKDirections(request:
mkDirectionsRequest)
        mkDirections.
        calculateDirectionsWithCompletionHandler {
            (response, error) -> Void in
                guard let response = response else {
                    if let error = error {
                        print("Error: \(error)")
                    }
                    return
                }
                let route = response.routes[0]
                self.mkMapView.addOverlay((route.polyline),
                    level: MKOverlayLevel.AboveRoads)
                let rect = route.polyline.boundingMapRect
                self.mkMapView.setRegion
                    (MKCoordinateRegionForMapRect(rect), animated:
true)
            } //End of calculateDirections
        } //End of viewDidLoad

        func mapView(mapView: MKMapView, rendererForOverlay
overlay: MKOverlay) -> MKOverlayRenderer {
            let mkPolylineRenderer =
                MKPolylineRenderer(overlay: overlay)
            mkPolylineRenderer.strokeColor =
                UIColor.redColor()
            mkPolylineRenderer.lineWidth = 4.0
            return mkPolylineRenderer
        }

        override func didReceiveMemoryWarning() {
            super.didReceiveMemoryWarning()
        }
    }
}
```

Besser wäre es – denn er ist vielleicht ortsfremd –, einen Weg vorzugeben. Dieser Weg sollte natürlich am besten direkt in der Karte sichtbar sein und berücksichtigen, ob der Anwender zu Fuß oder mit dem Auto unterwegs ist.

So erkennt der Benutzer neben den markierten Punkten auch gleichzeitig den besten Weg dorthin und kann mit Hilfe der Karte entsprechend planen.

Strecke im Projekt

Als Grundlage für ein solches Projekt wählt man am besten eine Vorlage vom Typ *Single View Application*. Im *ViewController* wird ein *MapView*-Control via Drag and Drop abgelegt. Da die Strecke etwas länger ist, wird die *MapView* mit Hilfe von *Constraints* über die komplette Fläche an den Seiten der *View* verankert (Bild 1).

Ein zu klein gewählter Kartenausschnitt ist dabei keine gute Wahl, denn die Übersicht geht in diesem Fall verloren. Anschließend wird mittels der *import*-Anweisung das *MapKit*-Framework eingebunden.

Die Vorbereitungen werden abgeschlossen, indem ein Outlet für das *MapView*-Control im Code via Drag and Drop erzeugt wird. Innerhalb der Klasse *ViewController* müssen noch zwei weitere Punkte für die ordnungsgemäße Funktion implementiert werden. Zunächst muss eine Ableitung des Protokolls *MKMapViewDelegate* in den Code eingefügt werden, und außerdem wird im Ereignis *ViewDidLoad* der Eigenschaft *delegate* des *MapView*-Controls das Schlüsselwort *self* zugewiesen.

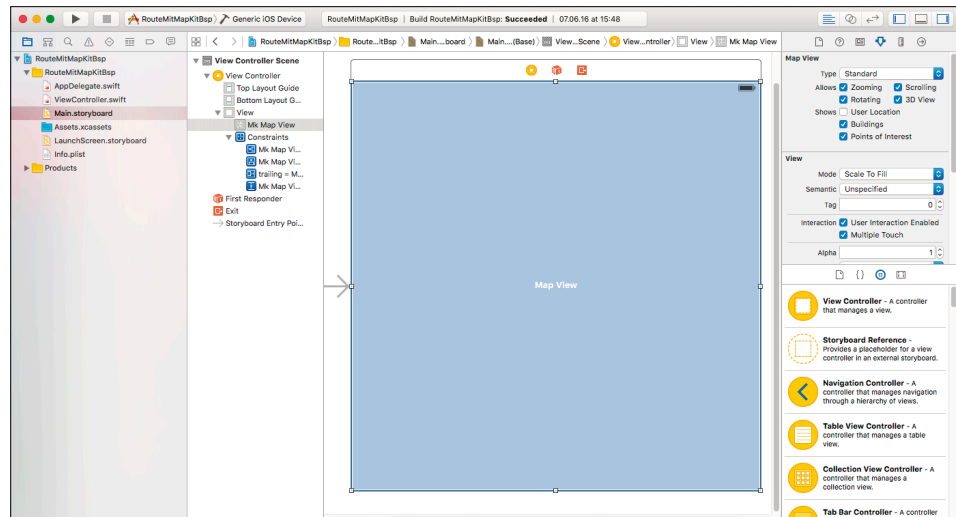
Source und Destination

Um eine Strecke innerhalb einer *MapView* einzeichnen zu können, bedarf es natürlich sowohl eines Start- als auch eines Endpunkts. Diese beiden Punkte werden in der App durch Instanzen der Klasse *CLLocationCoordinate2d* festgelegt. Dem Initialisierer der Klasse werden dabei die Koordinaten des Start- und des Endpunkts übergeben (Listing 1).

Damit die passenden Koordinaten übergeben werden können, muss man zuvor natürlich wissen, wohin es gehen soll. Im Beispiel soll eine Strecke vom Berliner Hauptbahnhof bis zum Brandenburger Tor eingezeichnet werden. Die beiden Variablen werden passenderweise mit *startPoint* beziehungsweise *endPoint* benannt.

Start- und Endmarkierung

Eine weitere Klasse wird zur Bestimmung von Start- und Endpunkt benötigt. Mit Instanzen der Klasse *MKPlacemark* können zusätzliche Informationen zu einem bestimmten Punkt gesichert werden, der über Koordinaten festgelegt wird. Beispielsweise sind dies: das Land, die Stadt und auch die Ad-



Die *MapView* wird mittels *Constraints* verankert (Bild 1)

resse. Die Klasse bietet hierfür auch gleich einen passenden Initialisierer an, der neben den Koordinaten zusätzlich noch ein Dictionary entgegennehmen kann, das Adressen aus dem Adressbuch enthält.

Im dargestellten Beispiel wird von dieser Möglichkeit allerdings kein Gebrauch gemacht. Es werden ausschließlich die Koordinaten übergeben. Der zweite Parameter wird mit *nil* vorbelegt.

Den beiden Variablen des Typs *MKPlacemark* wird die Bezeichnungen *startMKPlacemark* und *endMKPlacemark* gegeben. Aller guten Dinge sind bekanntlich drei. Nicht nur die Strecke soll in der Karte eingezeichnet werden, es soll auch je eine Markierung für den Start- und den Endpunkt in der Karte sichtbar sein.

Hierfür kommen Instanzen der Klasse *MKMapItem* zum Einsatz. Der Klasse wird zur Anlage der Instanzen jeweils die zuvor erzeugte *MKPlacemark*-Instanz übergeben. Wenn man die Symbole auf der Karte berührt, sollen diese auch preisgeben, welcher Punkt sich dahinter verbirgt. Hierfür werden wieder jeweils zwei Variablen benötigt, denen ein Text zugewiesen wird.

Erledigt wird das mittels der Klasse *MKPointAnnotation*. Den beiden Instanzen wird der Text *Hauptbahnhof* beziehungsweise *Brandenburger Tor* zugewiesen. Dies geschieht durch Zuweisung des Strings an die entsprechende Eigenschaft *title*. Zuletzt werden die beiden Instanzen von *MKPointAnnotation* mittels der Methode *showAnnotations* aktiviert.

Anweisungen für die Route

Die Berechnung der Route steht als Nächstes auf dem Programm. Zur Berechnung der Route wird auf die Klasse *MKDirectionsRequest* zurückgegriffen. Diese Klasse ermittelt anhand eines Start- und eines Endpunkts eine Strecke.

Zu diesem Zweck werden im Programm mehrere Eigenschaften der Klasse angesprochen. Der *source*-Eigenschaft wird der Startpunkt der Navigation zugewiesen. Hierbei muss es sich in jedem Fall um ein Objekt des Typs *MKMapItem* handeln. ►

Der Endpunkt wird der Eigenschaft *destination* zugewiesen. Auch hier muss der Typ *MKMapItem* verwendet werden. Im Beispiel entspricht das den zuvor angelegten Variablen: *startMkMapItem* und *endMkMapItem*.

Eine Information zur Berechnung der Route fehlt allerdings noch, und diese ist nicht unerheblich: Es muss angegeben werden, welches Transportmedium zur Berechnung der Strecke zugrunde gelegt werden soll, ob also die Strecke zu Fuß (*Walking*), mit dem Auto (*Automobile*) oder mit öffentlichen Verkehrsmitteln (*Transit*) zurückgelegt werden soll, oder ob es dem Benutzer egal ist (*Any*).

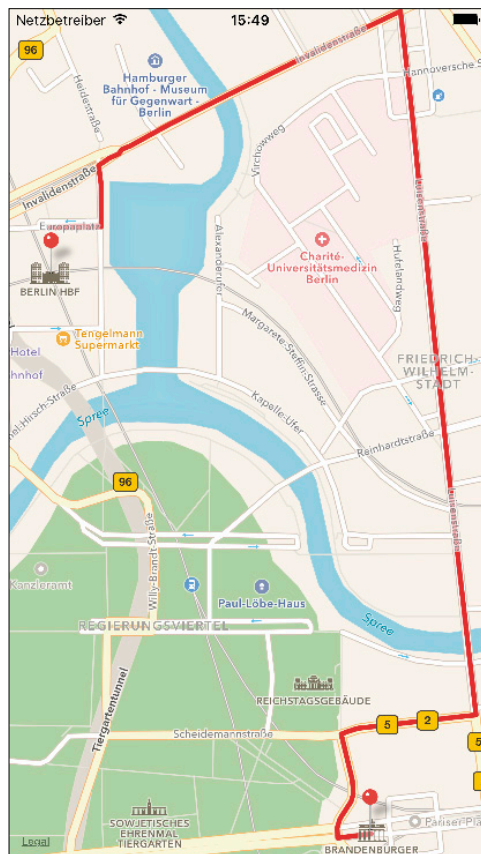
Verwaltet werden diese Typen in der Enumeration *MKDirectionsTransportType*. Dass der gewählte Typ Einfluss auf die berechnete Route hat, versteht sich von selbst. Im Beispiel wurde der Typ *Automobile* gewählt (**Bild 2**), was sich in der Berechnung der Strecke bemerkbar macht. Mit den festgelegten Informationen kann die Route nun berechnet werden. Hierfür wird die *MKDirectionsRequest*-Instanz einem Initialisierer der Klasse *MKDirections* übergeben.

Route berechnen

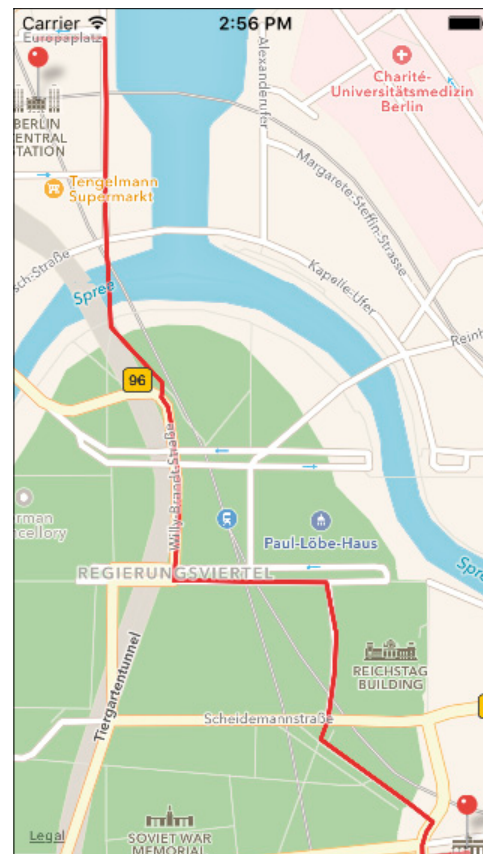
Der Anstoß zur Berechnung der Route erfolgt durch einen Aufruf der Methode *calculateDirectionsWithCompletionHandler*. Es handelt sich hierbei um einen asynchronen Aufruf. Innerhalb des Handlers wird der Code zur Bearbeitung des durch die Berechnung erstandenen Ergebnisses abgelegt. Hierzu wird die Verarbeitung durch eine *guard*-Anweisung gekapselt.

Tritt während der Berechnung ein Fehler auf, so wird dieser mit einer *print*-Anweisung ausgegeben. Im fehlerfreien Fall erfolgt mit der nachfolgenden Anweisung das Auslesen der Route, und sie wird in der Variablen *route* gespeichert. Damit die Route letztendlich gezeichnet werden kann, werden die ermittelten Informationen mittels der Methode *addOverlay* in die Karte übertragen. Sobald diese Daten vorliegen, wird die Methode *rendererForOverlay* aufgerufen.

In dieser Methode wird mit Hilfe einer Instanz vom Typ *MKPolyLineRenderer* die Route in die Karte gezeichnet. *MKPolyLineRenderer* selbst benötigt aber auch noch eine Basis-konfiguration. So muss über die Eigenschaft *strokeColor* die Farbe festgelegt werden, in der die Route gezeichnet wird,



Die berechnete Route zum Brandenburger Tor mit dem Auto (**Bild 2**)



Die berechnete Route zum Brandenburger Tor zu Fuß (**Bild 3**)

und auch die Strichstärke (*lineWidth*) kann auf diese Weise festgelegt werden.

Im Anschluss kann das Ergebnis der Karte entnommen werden. Soll statt des Autos ein Weg für einen Fußgänger berechnet werden, so muss nur der Enumerations-Typ *MKDirectionsTransportType* geändert werden – in diesem Fall auf *Walking*. Die Ansicht der Route innerhalb der Karte ändert sich dann natürlich, wie man auch **Bild 3** entnehmen kann.

Fazit

Die Arbeit mit dem iOS MapKit eröffnet Entwicklern zahlreiche Optionen bei der Integration von Karten in eigene Apps. Nicht nur die Markierung von POIs ist innerhalb einer Map-View-Instanz möglich. Auch das Einzeichnen einer Route ist nicht schwer und kann gut genutzt werden, um einem Anwender den Weg zum Ziel zu visualisieren. ■



Christian Bleske

ist Autor, Trainer und Entwickler mit dem Schwerpunkt Client/Server und mobile Technologien. Sein Arbeitsschwerpunkt liegt auf Microsoft-Technologien.
cb.2000@hotmail.de

Updates für Ihr Know-How

„Fortschritt heißt für mich vor allem, dass man fortschreiten will!“

Johannes Hoppe
IT-Berater, Programmierer, Webdesigner



Cross-Plattform-Apps mit C# und Xamarin

Trainer: Sebastian Seidel

3 Tage, 22.-24.11.2016, Köln
Ab 2.199,- EUR zzgl. MwSt.



Angular 2 mit TypeScript

Trainer: Johannes Hoppe, Gregor Woiwode

3 Tage, 21.-23.11.2016, Köln
Ab 2.199,- EUR zzgl. MwSt.



Effiziente Softwarearchitekturen mit PHP

Trainer: Stefan Priebisch

2 Tage, 14.-15.11.2016, München
Ab 1.799 EUR zzgl. MwSt.



Codequalität mit JavaScript

Trainer: Golo Roden

3 Tage, 02.-04.11.2016, Köln
Ab 2.199 EUR zzgl. MwSt.



Ihr Ansprechpartner:

Fernando Schneider – Key Account Manager – developer media

Telefon: +49 (0)89 74117-831 – E-Mail: fernando.schneider@developer-media.de

REACT-NATIVE-BIBLIOTHEKEN (TEIL 2)

Nativer Code trifft React Native

Die innovative Architektur von JavaScript-Frameworks der zweiten Generation erlaubt Entwicklern die Erstellung von Apps, die mit geringem Aufwand auf native APIs zugreifen.

Am Ende des ersten Teils der Artikelserie hatten wir eine fertige JavaScript-Applikation zusammengestellt, die die Möglichkeiten des Webteils von React Native demonstriert. In dieser Ausgabe ist es an der Zeit, Experimente mit dem für den Zugriff auf das Android-API vorgesehenen JavaScript-Java-Interface durchzuführen.

Wer mit Produkten wie PhoneGap Erfahrungen gesammelt hat, sollte sie an dieser Stelle verwerfen. Facebooks JavaScript-Interface ist nach Ansicht des Autors eine der besten Versionen am Markt – auch komplexe Interoperabilitäts-szenarien lassen sich mit minimalem Aufwand abdecken.

Die im vorigen Heft besprochenen Installationsanleitungen gelten auch an dieser Stelle. Wer ein neues Projekt anlegen möchte, kann dies durch Nutzung des *react-native*-Kommandos tun. Auch bei der Arbeit mit dem nativen Teil des Projekts ist es erforderlich, den für die Auslieferung von JavaScript zuständigen Server zu aktivieren. Dies lässt sich im Projektfenster durch Eingabe des Befehls *react-native-start* bewerkstelligen: Achten Sie darauf, dass der Server während des gesamten Entwicklungsprozesses laufen muss (Bild 1).

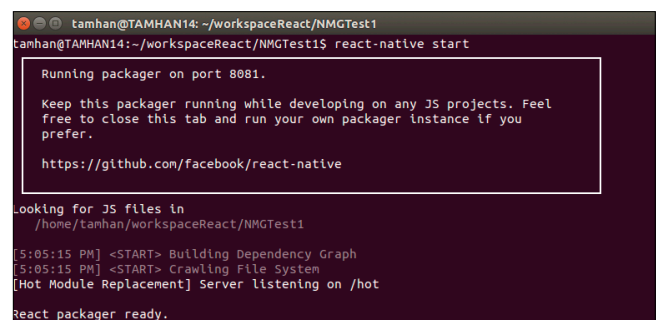
Android Studio starten

Starten Sie im nächsten Schritt Android Studio und öffnen Sie die Datei *build.gradle*. Sie enthält ein vollständiges Android-Studio-Beispielprojekt, das sich bei laufendem React-Native-Server durch Anklicken des Play-Buttons auf ein angeschlossenes Telefon übertragen lässt. Für die Kommunikation zwischen Telefon und Deployment-Server ist der im letzten Abschnitt im Detail besprochene Befehl *adb reverse* notwendig, der sich folgendermaßen aktivieren lässt:

```
tamhan@TAMHAN14:~/Android/Sdk/platform-tools$ ./adb
reverse tcp:8081 tcp:8081
```

Damit können wir uns auch schon der Realisierung des nativen Teils der Applikation zuwenden. Native Module sind in React Native immer von der Klasse *ReactContextBaseJavaModule* abgeleitet. Eine grundlegende Implementierung sieht folgendermaßen aus:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule {
    public NMGFirstModule(ReactApplicationContext
```



Dieser Server ist einsatzbereit (Bild 1)

```
reactContext) {
    super(reactContext);
}
@Override
public String getName() {
    return "NMGErstesModul";
}
}
```

Neben dem aufgrund des Aufbaus der Mutterklasse vorgeschriebenen Konstruktor findet sich hier auch eine Methode namens *GetName*. Sie liefert einen String zurück, unter dem das Java sprechende Modul später auf JavaScript-Seite ansprechbar ist.

Module exponieren normalerweise eine oder mehrere Funktionen. Dies geschieht durch Nutzung des *ReactMethod*-Makros, das nach folgendem Schema vor die Deklaration der zu exponierenden Methode platziert wird:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule {
    ...
    @ReactMethod
    public void show(String message) {
        Log.e("NMG",message)
    }
}
```

Die von React Native exponierten Funktionen haben als Rückgabtyp immer *Void*. Das liegt daran, dass die Java-

JavaScript-Schnittstelle asynchron arbeitet: Wer Werte zurückgeben möchte, muss dies über Callbacks bewerkstelligen. Für unsere jetzigen Bedürfnisse reicht es allerdings aus, den angelieferten String entgegenzunehmen und anschließend durch Nutzung der *Log*-Klasse in Richtung von Android Studio zu senden.

Containerklasse ReactPackage

Eine Ebene darüber findet sich das *ReactPackage*. Es handelt sich dabei um eine Containerklasse, die verschiedene native Module auf eine kompakte und bequeme Art und Weise zusammenfasst. Die für unser Projekt erforderliche Basisimplementierung zeigt [Listing 1](#).

Neben den hier nicht benötigten Methoden zum Anmelden von JavaScript-Modulen und Steuerelementverwalten erlaubt *createNativeModules* das Anmelden von nativen Modulen wie dem im vorigen Schritt erstellten *NMGFirstModule*. Achten Sie darauf, dass nicht benötigte *get*-Methoden eine leere Liste zurückliefern müssen. Das von Android Studio vorgeschlagene Retournieren von *null* führt zu undefiniertem Verhalten.

Zu guter Letzt müssen wir das neue Modul noch bei der Engine anmelden. Dies erfolgt im Einsprungpunkt der Applikation:

```
public class MainActivity extends ReactActivity
{
    ...
    @Override
    protected List<ReactPackage> getPackages()
    {
        return Arrays.<ReactPackage>asList(
            new MainReactPackage(),
            new NMGFirstModulePackage());
    }
}
```

getPackages liefert nun neben dem schon bekannten *MainReactPackage*-Paket auch das soeben erstellte *NMGFirstModulePackage* zurück, das so für die JavaScript-Runtime ansprechbar gestaltet wird.

Damit sind die Arbeiten auf Seiten des nativen Programms abgeschlossen. Wer mit der Applikation aus der vorigen Heft-

Listing 1: Basis-Implementierung

```
public class NMGFirstModulePackage implements
ReactPackage {
    @Override
    public List<NativeModule> createNativeModules
(ReactApplicationContext reactContext) {
        List<NativeModule> modules = new
        ArrayList<>();
        modules.add(new NMGFirstModule
        (reactContext));
        return modules;
    }
    @Override
    public List<Class<? extends JavaScriptModule>>
    createJSModules() {
        return Collections.emptyList();
    }
    @Override
    public List<ViewManager> createViewManagers
(ReactApplicationContext reactContext) {
        return Collections.emptyList();
    }
}
```

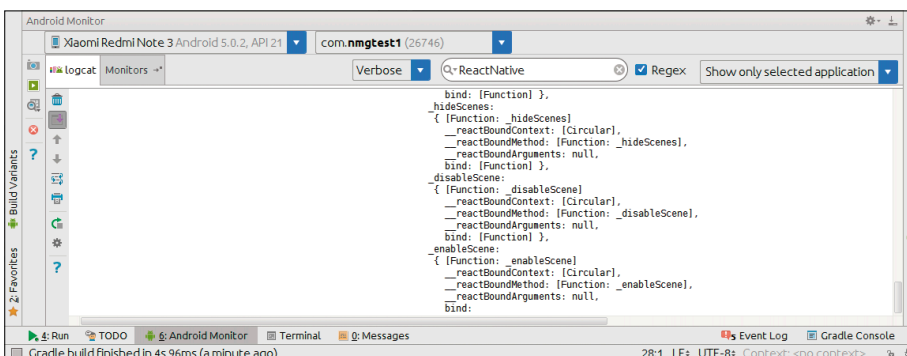
ausgabe weiterarbeitet, öffnet an dieser Stelle die Datei *listpage.js* und erweitert die Render- und Konstruktormethoden nach folgendem Schema:

```
import React, { Component } from 'react';
...
import { NativeModules } from 'react-native';
```

Im Moment wird unser Modul bei Bedarf aus der *NativeModules*-Klasse hervorgeholt. Wer dieses Problem umgehen möchte, kann einen JavaScript-Wrapper realisieren. Nach Ansicht des Autors ist der dazu notwendige Aufwand allerdings nur in den wenigsten Fällen gerechtfertigt.

Der eigentliche Aufruf erfolgt dann wie bei einem normalen Framework-Objekt. Der String *NMGErstesModul* wurde weiter oben in *GetName* festgelegt:

```
export class ListPage extends
Component {
    constructor(props) {
        super(props);
        NativeModules.NMGErstesModul.
        show("Konstruktor
        aufgerufen");
        ...
    }
    render(){
        NativeModules.
        NMGErstesModul.
        show("Render aufgerufen");
        ...
    }
}
```



Keine Anzeige in Android Studio (Bild 2)

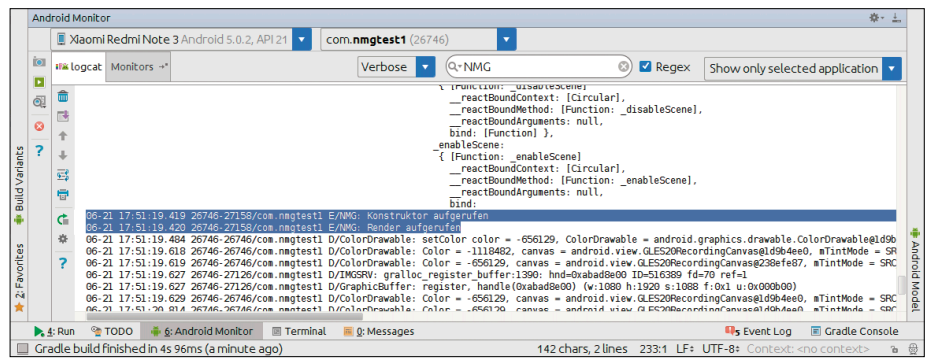
Damit ist die App auch schon zur Ausführung bereit. Geben Sie `./adb reverse tcp:8081 tcp:8081` ein, um das Smartphone mit der Workstation zu verbinden und den Zugriff auf den Code-Server zu ermöglichen. Starten Sie das Projekt dann durch Anklicken von *Play*.

Im Monitor-Tab von Android Studio erscheint allerdings nichts (Bild 2). Dieses seltsame Verhalten liegt daran, dass Android Studio von Haus aus den Applikationsnamen als Filter einsetzt. Wer stattdessen den Filterstring einstellt, sieht die beiden Aufrufe der Log-Methode (Bild 3).

Überladung und andere Experimente

Java unterscheidet sich insofern von JavaScript, als erstere Sprache streng typisiert ist. So kennt Java mehrere unterschiedliche Zahlentypen, während man in JavaScript nur mit *Number* auskommen muss. Diese Situation animiert uns zur Realisierung eines Experiments. Ergänzen Sie das NMGFirstModule um die in Listing 2 gezeigten vier Methoden, die bei identischem Namen verschiedene Parametertypen aufweisen und für Java eindeutig unterscheidbar sind.

Schon auf den ersten Blick fällt auf, dass die JavaScript-Runtime an dieser Stelle ein Problem bekommt. Die Funktion *doSomething* liegt nämlich in vier verschiedenen Varianten



Das native Modul funktioniert problemlos (Bild 3)

vor, die sich nur in der Art des übernommenen Elements unterscheiden. Überladungen sollten in der Praxis immer eindeutig sein: Wenn *byte* und *int* zu wesentlich unterschiedlichem Verhalten führen, sind Fehlbedienungen des Interfaces beziehungsweise der Klasse unvermeidbar.

Als Testharnisch für das Modul bietet sich abermals *ListPage.js* an, wo *NativeModules* nun mit folgendem Baum aufgerufen wird:

```
export class ListPage extends Component {
  constructor(props) {
    super(props);
    NativeModules.NMGErstesModul.doSomething(22);
    NativeModules.NMGErstesModul.doSomething(22222);
    NativeModules.NMGErstesModul.doSomething(22.22);
    NativeModules.NMGErstesModul.doSomething("22");
    NativeModules.NMGErstesModul.doSomething("ABCDE");
  }
}
```

Wer diesen Code unverändert ausführt, wird im ersten Schritt mit der in Bild 4 gezeigten Fehlermeldung konfrontiert. Dies liegt daran, dass der Parameter *Byte* in React Natives Java-JavaScript-Interface nicht vorgesehen ist. Im Bereich der Zahlenwerte werden nur die folgenden Parameter unterstützt: *Boolean* (Bool), *Integer* (Number), *Double* (Number) und *Float* (Number). Für die Handhabung nichtnumerischer Werte steht neben dem Datentyp *String* auch die Möglichkeit zur Verfügung, Callbacks zu verarbeiten. Zu guter Letzt gibt es noch zwei dedizierte Hilfsklassen, die zwischen JavaScript und Java makeln und die Übertragung komplexer Objekte erleichtern – *ReadableMap* und *ReadableArray*.

Wer sich an dieser Stelle versucht fühlt, die *Byte*-Methode zu entfernen, erhält nach wie vor einen Fehler als Resultat (Bild 5). Reduziert man den Inhalt der Klasse auf eine Methode mit dem *String*-Parameter, lässt aber den Aufruf von *doSomething* unverändert, so bekommt man stattdessen den Fehler in Bild 6. Kurz gefasst: Methodenüberladung ist in React Native nicht möglich. Wer in seinem schon vorhandenen Java-Code im Moment auf überladene Methoden setzt, muss umschreiben.

Zustandshaltung leicht gemacht

NativeScript erlaubt Entwicklern das Instanzieren von nativen Objekten, die auf JavaScript-Seite über eine Dummy-

Listing 2: Vier Methoden

```
public class NMGFirstModule extends
  ReactContextBaseJavaModule {
  ...
  @ReactMethod
  public void doSomething(String _something)
  {
    Log.e("NMG", "_something + "als String");
  }
  @ReactMethod
  public void doSomething(byte _something)
  {
    Log.e("NMG", "_something + "als Byte");
  }
  @ReactMethod
  public void doSomething(int _something)
  {
    Log.e("NMG", "_something + "als Integer");
  }
  @ReactMethod
  public void doSomething(float _something)
  {
    Log.e("NMG", "_something + "als Float");
  }
}
```

Klasse referenziert werden. In React Native ist dies nicht möglich: Die Zustandshaltung kann entweder auf JavaScript- oder auf Java-Seite erfolgen.

In zweitgenannten Fall bietet sich die Verwendung eines an OpenGL erinnernden Designparadigmas an. Zur Erinnerung: Ressourcen werden im ersten Schritt bei der Engine angemeldet. Diese übernimmt die Informationen daraufhin in ihren eigenen Speicher und gibt dem Aufrufer eine Zahl zurück.

Zukünftige Invokationen übergeben einfach diesen Wert, wenn sie auf die betreffende Ressource hinweisen wollen. Wir wollen nun unser Beispiel um eine Klasse erweitern, die als Backend dient. Dazu implementieren wir ein Singleton-Pattern:

```
public class NMGStorageSingleton {
    static NMGStorageSingleton mySingleton=
        new NMGStorageSingleton();
    private NMGStorageSingleton() {
        myStore=new HashMap<Integer, String>();
        myCounter=0;
    }
    public static NMGStorageSingleton getInstance() {
        return mySingleton;
    }
}
```

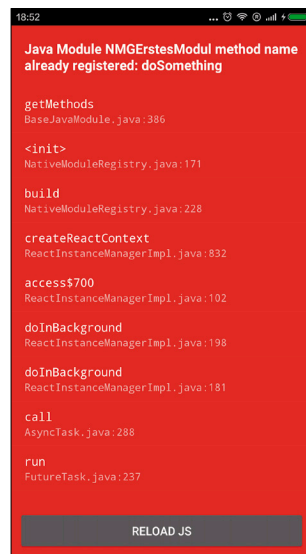
Neben der eigentlichen Implementierung des Design-Patterns benötigen wir einen ID-Generator, der die IDs sequenziell generiert:

```
public int getID() {
    return myCounter++;
}
```

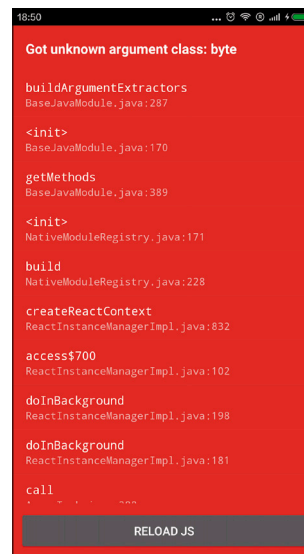
Zu guter Letzt ist noch ein Speicher notwendig, der die einzelnen Instanzen der zu verwaltenden Daten – wir beschränken uns hier auf Strings – vorhält. Auch er wird als Member-Variable des Singletons angelegt und steht somit während der gesamten Programmlaufzeit zur Verfügung:

```
int myCounter;
public HashMap <Integer, String> myStore;
```

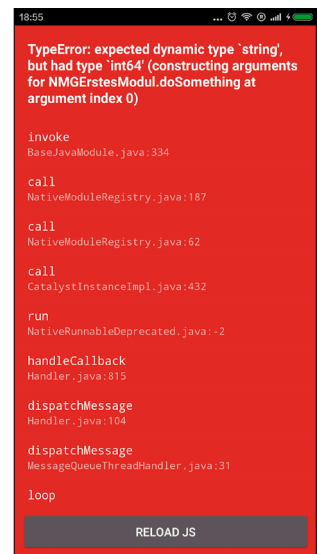
Als nächste Aufgabe müssen wir im nativen Plug-in eine Methode deklarieren, die das Anlegen eines Strings samt dem Zurückgeben seiner ID ermöglicht. Die Kommunikation zwischen Java und ReactNative erfolgt prinzipiell unter Nutzung



Parameter vom Typ byte sind nicht populär (Bild 4)



Überladungen finden bei React Native auch nur wenig Gnade (Bild 5)



Zahlenwerte konvertiert React Native nicht automatisch in Strings (Bild 6)

von Callbacks, weshalb der Methodenkörper folgendermaßen aussieht:

```
public class NMGFirstModule extends
    ReactContextBaseJavaModule {
    ...
    @ReactMethod
    public void saveAString(String _whichOne,
        Callback _returnSuccess) {
        int id;
        NMGStorageSingleton mySingleton =
            NMGStorageSingleton.getInstance();
        id=mySingleton.getID();
        mySingleton.myStore.put(id,_whichOne);
        _returnSuccess.invoke(id);
    }
}
```

Neben der Instanziierung des Singletons findet sich hier auch die Aktivierung des Callback-Objekts: Seine *invoke*-Methode nimmt die Parameter entgegen, die in Richtung JavaScript wandern sollen. Achten Sie darauf, dass zur Kompilationszeit keine Verifikation des Parametersatzes erfolgen kann. Eine Reflexion des Callback-Interfaces demonstriert dies:

```
package com.facebook.react.bridge;
public interface Callback {
    public void invoke(Object... args);
}
```

Das auf Seiten von JavaScript erfolgende Realisieren des Callbacks lässt sich beispielsweise so erledigen:

```
export class ListPage extends Component {
    constructor(props) {
        super(props);
    }
}
```

```
NativeModules.NMGErstesModul.
saveAsString("Variable A", (id) =>
{ console.log(id); },);
...
```

Da die Syntax mit Klammern nicht jedermanns Geschmack ist, bietet sich alternativ natürlich auch die Nutzung klassischer Funktionen an:

```
export class ListPage extends
Component {
  idComingIn(what){
    console.log(what);
  }
  constructor(props) {
    super(props);
    NativeModules.NMGErstesModul.saveAsString
    ("Variable ", this.idComingIn);
    ...
  }
}
```

Zum Test dieses Programms können Sie mehrere Einträge hintereinander anlegen und im Debugger einen Breakpoint setzen. Der in **Bild 7** gezeigte Screenshot entstand durch folgendes Snippet:

```
constructor(props) {
  super(props);
  NativeModules.NMGErstesModul.saveAsString("Variable A",
(id) => { console.log(id); });
  NativeModules.NMGErstesModul.saveAsString("Variable B",
this.idComingIn);
  NativeModules.NMGErstesModul.saveAsString("Variable C",
(id) => { console.log(id); });
  ...
}
```

Die Arbeitsweise, einen Breakpoint direkt im nativen Teil zu platzieren, ist in React Native weder unüblich noch problematisch. Das Deployment der Ausführungsumgebung erfolgt ja von Android Studio aus. Die IDE kann das von ihr losgetretene Programm natürlich jederzeit anhalten.

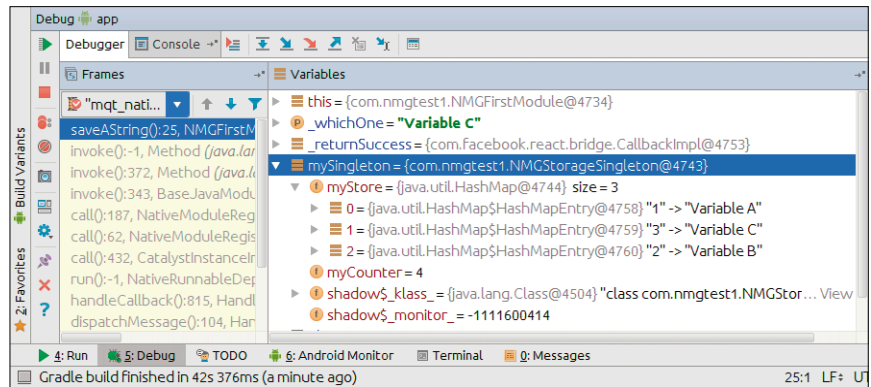
Promises im Einsatz

Microsofts asynchrones C# spielt mittlerweile auch in der Welt von JavaScript eine Rolle. Der ECMAScript-Standard implementiert das Konzept der Promises.

Es handelt sich dabei um eine Klasse, die eine beliebige, im Hintergrund ablaufende und potenziell blockierende Operation enkapsuliert. Die Besonderheit besteht darin, dass die Abarbeitung des normalerweise asynchron ablaufenden Jobs über den *await*-Operator abgewartet werden kann.

Zur Implementierung von Promises muss im nativen Modul eine nach dem folgenden Schema aufgebaute Methode platziert werden:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule {
```



Die Elemente sind am Platz (Bild 7)

```
@ReactMethod
public void returnAField(Promise _field) {
  _field.resolve("Hallo Promise!");
}
```

Die in React Native implementierte *Promise*-Klasse ist im Großen und Ganzen friedfertig. Die einzige wichtige Regel bei ihrer Nutzung ist, dass das Promise immer an letzter Stelle in der Parameterliste stehen muss – ist dies nicht der Fall, kommt es zu undefiniertem Programmverhalten.

ReturnAField nutzt die Methode *resolve*, um die erfolgreiche Abarbeitung der von Promise abgebildeten Arbeit zu bestätigen. Als Alternative dazu gibt es einige Ablehnungsmethoden, die durch Reflexion der Promise-Klasse folgendermaßen aufgeschlüsselt werden:

```
public interface Promise {
  void resolve(@Nullable Object value);
  void reject(String code, String message);
  void reject(String code, Throwable e);
  void reject(String code, String message, Throwable e);
  void reject(Throwable reason);
}
```

Die eigentliche Implementierung der soeben realisierten Funktion auf Seiten des Java-Codes erfolgt nach der vom normalen Promise bekannten Vorgehensweise:

```
export class ListPage extends Component {
  async myRunner() {
    myData = await
    NativeModules.NMGErstesModul.returnAField();
    NativeModules.NMGErstesModul.show(myData);
  }
  constructor(props) {
    super(props);
    this.myRunner();
  }
}
```

myRunner() wirkt auf den ersten Blick überflüssig. Dies ist allerdings nicht der Fall – der ECMAScript-Standard schreibt vor, dass *await*-Befehle nur in als *async* gekennzeichneten Methoden vorkommen dürfen. Die folgende – kompaktere –

Implementierung würde zum in **Bild 8** gezeigten Fehlerbild führen:

```
export class ListPage
extends Component {
  constructor(props) {
    super(props);
    myData = await
    NativeModules.
    NMGErstesModul.
    returnAField();
    NativeModules.
    NMGErstesModul.
    show(myData);
  }
}
```

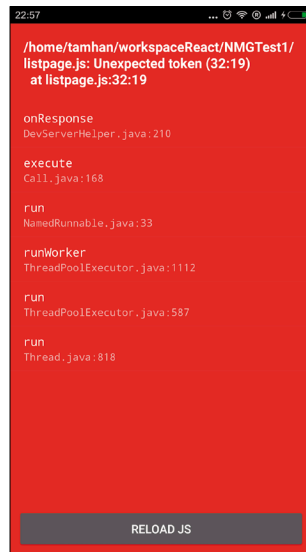
Aus reinem Interesse wollen wir an dieser Stelle noch das Ablehnen eines Promise vorstellen. Dazu muss im nativen Modul eine weitere Methode angelegt werden, die die Ausführung des Promise durch Nutzung einer der *reject*-Funktionen ablehnt:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule {
  ...
  @ReactMethod
  public void rejectAField(Promise _field) {
    _field.reject("NMG-ERROR",
    "Grüßen des Promise schlug fehl");
  }
}
```

Aufseiten der JavaScript-Applikation wird die Exception dann folgendermaßen entgegengenommen:

```
export class ListPage extends Component {
  async myRunner(){
    try{
      await NativeModules.NMGErstesModul.
      rejectAField();
    }
    catch (e){
      console.error(e);
    }
  }
}
```

Wer das Programm aus Android Studio heraus anwirft, sieht die in **Bild 9** gezeigte Statusmeldung: Abgelehnte Promises



Die Nutzung von await in Nicht-async-Funktionen führt zu seltsamen Fehlern (Bild 8)

können dem JavaScript-Teil des Codes eine Vielzahl von Informationen über die Ursache des Fehlschlags anliefern.

Fortgeschrittene Maklerei

Das Austauschen von einzelnen Primitivwerten mag im kleinen Beispiel funktionieren. Ab einer gewissen Codegröße wäre es jedoch wünschenswert, wenn man ganze Arrays beziehungsweise Objekte zwischen den beiden Sprachen hin- und hersenden könnte.

Bevor wir uns mit der praktischen Implementierung beschäftigen, noch ein allgemeiner Hinweis. Die zwischen den beiden Sprachen hin- und hergesendeten Objekte können keine Memberfunktionen enthalten: Ein zwischen Java und JavaScript hin- und hergeschicktes Objekt ist im Grunde genommen ein Array, das mit Strings indiziert wird.

Dieses Verhalten lässt sich am einfachsten durch Betrachtung des Readable-Map-Interfaces erklären. Die Methoden *hasKey* und *isNull* erlauben dem Entwickler das Feststellen, ob ein bestimmter String mit einem Wert verbunden ist:

```
public interface ReadableMap {
  boolean hasKey(String name);
  boolean isNull(String name);
}
```

Die *get**-Methoden liefern im Fall eines vorhandenen Elements die dort gespeicherten Werte zurück:

```
public interface ReadableMap {
  ...
  boolean getBoolean(String name);
  double getDouble(String name);
  int getInt(String name);
  String getString(String name);
  ReadableArray getArray(String name);
  ReadableMap getMap(String name);
  ReadableType getType(String name);
}
```

Eine Sonderrolle spielt *getType* insofern, als die Methode über die Art der unter diesem String vorliegenden Informationen informiert. Zum Zeitpunkt der Drucklegung unterstützt Facebook folgende Parametertypen:

```
public enum ReadableType {
  Null, Boolean, Number, String, Map, Array,
}
```

Mit diesem Wissen können wir uns einer Funktion zuwenden, die eine Map entgegennimmt. Um unser Beispiel etwas interessanter zu machen, wollen wir eine einfache arithmeti- ►



Die zurückgegebene Fehlermeldung ist vergleichsweise komplex (Bild 9)

sche Operation realisieren. *returnAndTakeAMap* nimmt ein Objekt mit zehn benannten Feldern entgegen und errechnet die Summe aller bevölkerten Schlüssel:

```
@ReactMethod
public void returnAndTakeAMap(ReadableMap _aMap,
    Callback _myAnswer) {
    String myIndexRunner;
    int myAccumulator=0;
    for(int i=0;i<10;i++) {
        myIndexRunner="idx"+i;
        if(_aMap.containsKey(myIndexRunner) && _aMap.getType
            (myIndexRunner)==ReadableType.Number) {
            myAccumulator+=_aMap.getDouble
                (myIndexRunner);
        }
    }
    _myAnswer.invoke(myAccumulator);
}
```

Neben der Nutzung der im vorigen Schritt besprochenen Methoden von *Readable-Map* ist hier eigentlich nur das Vorhandensein eines Callbacks interessant, das zum Zurückliefern des Ergebnisses dient.

Aufseiten des JavaScript-Codes entsteht eine *Readable-Map* als ein gewöhnliches Objekt. Ein Beispiel für den Aufruf unserer Methode sähe folgendermaßen aus:

```
export class ListPage extends Component {
    constructor(props) {
        super(props);
        myElement=new Object();
        myElement.idx1=99;
        myElement.idx3=99;
        myElement.idx5=99;
        NativeModules.NMGErstesModul.returnAndTakeAMap
            (myElement, (id) => { console.log(id); });
        ...
    }
}
```

Maps lassen sich nicht nur zum Entgegennehmen von Daten nutzen. Der Java-Code kann eine Map auch mit Parametern bevölkern, um sie im nächsten Schritt in Richtung des JavaScript-Teils zu verschieben. Im Fall von *returnAndTakeAMap* sind dazu folgende Änderungen notwendig:

```
@ReactMethod
public void returnAndTakeAMap(ReadableMap _aMap,
    Callback _myAnswer) {
    ...
    WritableMap returnableMap= Arguments.createMap();
    returnableMap.putInt("aresult", myAccumulator);
    _myAnswer.invoke(returnableMap);
}
```

WritableMap ist insofern interessant, als die Klasse keinen klassischen New-Constructor aufweist. Die Erzeugung einer neuen Instanz erfolgt stattdessen durch das globale Argu-

ment-Objekt, das eine dedizierte Generatorfunktion anbietet. Für die Bevölkering mit Informationen steht derweil eine Gruppe mit *put*-Methoden zur Verfügung. Zum Zeitpunkt der Drucklegung lieferte eine Reflexion der Klasse das folgende Ergebnis:

```
public interface WritableMap extends ReadableMap {
    void putNull(String key);
    void putBoolean(String key, boolean value);
    void putDouble(String key, double value);
    void putInt(String key, int value);
    void putString(String key, String value);
    void putArray(String key, WritableArray value);
    void putMap(String key, WritableMap value);
    void merge(ReadableMap source);
}
```

Damit sind wir mit dem Java-Teil der *WritableMap* fertig. Das Entgegennehmen in JavaScript erfolgt nach folgendem Schema:

```
export class ListPage extends Component {
    constructor(props) {
        super(props);
        myElement=new Object();
        myElement.idx1=99;
        myElement.idx3=99;
        myElement.idx5=99;
        NativeModules.NMGErstesModul.returnAndTakeAMap
            (myElement, (id) => { console.log(id.aresult); });
    }
}
```

Schon aus Gründen der Vollständigkeit sei hier noch auf *WritableArray* und *ReadableArray* hingewiesen. Die beiden Datenstrukturen verhalten sich im Großen und Ganzen wie ihre Vorbilder auf Map-Basis. Der einzige Unterschied besteht darin, dass anstelle von Strings sequenzielle Zahlen zum Einsatz kommen. Dies lässt sich am Einfachsten beweisen, indem man die Grundstruktur der Klasse durch Reflexion auf den Bildschirm holt:

```
public interface ReadableArray {
    int size();
    boolean isNull(int index);
    boolean getBoolean(int index);
    double getDouble(int index);
    int getInt(int index);
    String getString(int index);
    ReadableArray getArray(int index);
    ReadableMap getMap(int index);
    ReadableType getType(int index);
}
```

Die Dokumentation von React Native lässt stellenweise zu Wünschen übrig. Der unter <https://github.com/facebook/react-native/tree/master/ReactAndroid/src/main/java/com/facebook/react/bridge> einsehbare Quellcode der Bridgeklassen ist auf jeden Fall lesenswert.

React Native lebt normalerweise in einer Activity. Die im Backend stehende Java-Engine kann in vielen Fällen davon profitieren, wenn sie sich an die Lifecycle-Events dieser Activity anhängen kann.

In React Native wird dies über eine Instanz des Interfaces *LifecycleEventListener* realisiert, die über *Implements* in das Modul eingebunden und im Rahmen der Initialisierung des Moduls beim Context angemeldet wird:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule implements
LifecycleEventListener {
    public NMGFirstModule(ReactApplicationContext
    reactContext) {
        super(reactContext);
        reactContext.addLifecycleEventListener(this);
    }
}
```

In *LifecycleEventListener* sind drei Methoden implementiert, deren Vorhandensein für eine erfolgreiche Implementierung erforderlich ist:

```
public class NMGFirstModule extends
ReactContextBaseJavaModule implements
LifecycleEventListener {
    @Override
    public void onHostResume() {
    }
    @Override
    public void onHostPause() {
    }
    @Override
    public void onHostDestroy() {
    }
}
```

Kenner des Android-Activity-Lifecycles kennen die drei Methoden mit Sicherheit aus dem in **Bild 10** gezeigten Diagramm. Die in React Native implementierten Methoden verhalten sich im Großen und Ganzen so, wie man es vom nativen Android erwarten würde.

Steuerelemente transformieren

Ein besonders interessantes Feature von React Native ist die Möglichkeit, beliebige native Steuerelemente über einen Wrapper für den XML-Parser ansprechbar zu machen. Da gewöhnliche Beispiele mit den in Android enthaltenen Steuerelementen langweilig sind, wollen wir an dieser Stelle stattdessen auf die von Philipp Jahoda entwickelte Charting-Bibliothek *MPAndroidChart* zurückgreifen.

Ihre Einbindung erfolgt streng nach Dokumentation – durch Änderungen in den Gradle-Buildfiles. Als Erstes ist die Version *Build.gradle* (Project: *NMGTest1*) erforderlich, deren Repositories-Block folgendermaßen angepasst wird:

```
allprojects {
    repositories {
```

```
mavenLocal()
jcenter()
maven {
    url "$projectDir/../../node_modules/
    react-native/android"
}
maven { url "https://jitpack.io" }
}
```

Die auf den ersten Blick ungewöhnliche Syntax ist korrekt. Es ist erlaubt, mehrere Maven-Blöcke in einer Repository-Struktur einzubinden. In der zum eigentlichen Applikationsprojekt gehörenden Build-Datei *build.gradle* (Module: *app*) wird der Dependencies-Baum dann so erweitert:

```
dependencies {
    compile fileTree(dir: "libs", include: ["*.jar"])
    compile "com.android.support:appcompat-v7:23.0.1"
    compile "com.facebook.react:react-native:+"
    compile 'com.github.PhilJay:MPAndroidChart:v2.2.5'
}
```

Android Studio wird beim Speichern der Gradle-Files ein Fly-out anzeigen, das zur Resynchronisation des Projekts auffordert. In der Praxis kommt es dabei normalerweise zu einem Fehler. Als erste Maßnahme zur Behebung empfiehlt sich ein Neustart der IDE. Klicken Sie daraufhin auf *Build* und *Rebuild project* und prüfen Sie, ob die Fehlermeldung verschwindet. Ist dies nicht der Fall, so schafft das auf der Unterseite befindliche Terminalfenster Abhilfe.

Öffnen Sie es und geben Sie den Befehl *./gradlew tasks* ein. Er schafft das Herunterladen der *.pom*-datei auch dann, wenn der normale Parser scheitert.

Auf Seiten des Java-Codes ist für eigene Steuerelemente eine Klasse erforderlich, die von *SimpleViewManager* abgeleitet wird. Die von Facebook bereitgestellte Basisklasse nimmt einen Template-Parameter entgegen, der den Typ des zu verwaltenden Steuerelements beschreibt:

```
public class ChartViewManager extends SimpleViewManager
<com.github.mikephil.charting.charts.LineChart> {
    @Override
    public String getName() {
        return "NMGLineChart";
    }
}
```

In ViewManagern sind zwei Methoden erforderlich: Erstens die Funktion *GetName*, die wie im vorhergehenden Fall für das Zurückgeben eines Strings zuständig ist, unter dem das betreffende Modul später auf JavaScript-Seite ansprechbar ist. Die Methode *createViewInstance* hat sodann die Aufgabe, bei Bedarf eine belebte Instanz des Steuerelements zurückzugeben:

```
@Override
protected com.github.mikephil.charting.charts.
```

```

LineChart createViewInstance(ThemedReactContext
reactContext) {
    LineChart myChart=new LineChart(reactContext);
    return myChart;
}

```

Die Anmeldung des ViewManagers erfolgt abermals im `ReactPackage`. Diesmal ist dazu die Methode `createViewManagers` erforderlich.

Damit können wir abermals auf die JavaScript-Seite wechseln. Im Header muss das Steuerelement bereitgestellt werden. Dazu ist die Methode `requireNativeComponent` erforderlich, die vor der Nutzung aus dem `react-native`-Paket per `require()` bereitgestellt werden muss. Der eigentliche Aufruf nimmt sodann den an `getName` übergebenen String entgegen, um die Korrelation zu ermöglichen:

```

import {
    AppRegistry,
    ...
    TouchableNativeFeedback,
    requireNativeComponent
} from 'react-native';
var NMGLineChart=requireNativeComponent
("NMGLineChart",null);

```

Dieser Teil des Codes ist insbesondere für Umsteiger von älteren Versionen von React Native kritisch: `requireNativeComponent` war früher im React-Hauptobjekt domiziliert. Die eigentliche Einbindung in Render hingegen läuft im Großen und Ganzen ohne Auffälligkeiten ab. Achten Sie nur darauf, dass das in der Methode verwendete Tag dem Namen der per `requireNativeComponent` angelegten Variable entspricht.

Programmatische Eigenschaften

Die von Facebook ausgelieferten Steuerelemente erlauben dem Entwickler die Beeinflussung ihres Verhaltens über an HTML erinnernde Eigenschaften. Im Moment steht diese Funktion bei unserem Charting-Steuerelement nicht zur Verfügung. Die anzuzeigenden Daten könnten über ein Singleton bereitgestellt werden, dessen Inhalte im Rahmen der Initialisierung der `ChartView` ausgelesen werden.

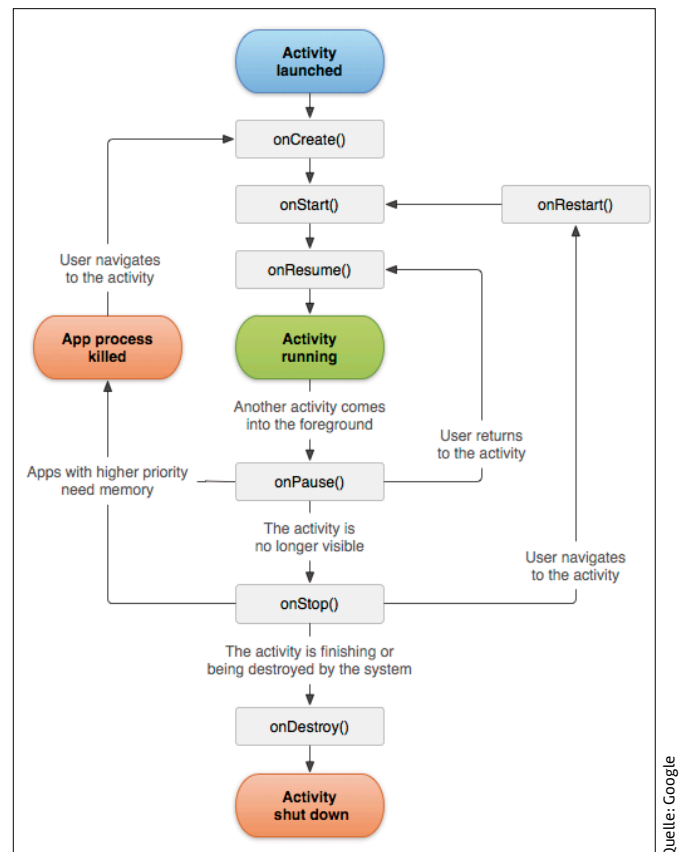
Eigene Properties lassen sich in ViewManagern durch Nutzung des `ReactProp`-Makros anlegen. Die Grundversion einer Eigenschaft präsentiert sich folgendermaßen:

```

public class ChartViewManager extends SimpleViewManager
<com.github.mikephil.charting.charts.LineChart> {
    @ReactProp(name="cdata")
    public void setData(com.github.mikephil.charting.
charts.LineChart _view, ReadableArray _cdata)
    {
    }
}

```

Im an `name` übergebenen String wird der Name der Eigenschaft festgelegt, unter dem sie später aufseiten von JavaScript ansprechbar ist. Unter dem Makro muss immer die De-



Quelle: Google

Ein Gutteil des Activity-Lebenszyklus ist auch in React Native ansprechbar (Bild 10)

klaration einer Settermethode folgen, die neben dem zu schreibenden Wert auch einen Verweis auf die zu verarbeitende Steuerelementinstanz entgegennimmt.

Da unser Beispiel einen komplexen Datentyp verarbeitet, gibt es keine Möglichkeit zum Festlegen eines Defaultwerts. Bei der Arbeit mit numerischen Informationen können Sie Standardparameter festlegen, die beim Fehlen des jeweiligen Attributs angenommen werden. Dazu dienen vier `default`-Funktionen, die folgendermaßen deklariert sind:

```

double defaultDouble() default 0.0;
float defaultFloat() default 0.0f;
int defaultInt() default 0;
boolean defaultBoolean() default false;

```

Ihre Nutzung erfolgt dann durch Bevölkerung des `ReactProp`-Makros, das auch mehrere durch Komma getrennte Parameter entgegennehmen kann:

```

@ReactProp(name = "borderRadius", defaultFloat = 0f)
public void setBorderRadius(ReactImageView view, float
borderRadius) {
    view.setBorderRadius(borderRadius);
}

```

Nach diesem kurzen Exkurs können wir zur Belegung des Diagramms schreiten. Per `MPAndroidChart` realisierte Line-

charts beziehen die anzuzeigenden Datenpunkte prinzipiell aus einer *ArrayList*, die ein oder mehrere auf der *Entry*-Klasse basierende Elemente anliefert. Leider ist unser *ReadableArray* mit *Entry* nicht kompatibel, weshalb wir die angelieferten Informationen in *Entries* konvertieren:

```
@ReactProp(name="cdata")
public void setData(com.github.mikephil.charting.charts.
LineChart _view, ReadableArray _cdata) {
    ArrayList<Entry> entries = new ArrayList<>();
    ArrayList<String> xValList=new ArrayList<>();
    for(int i=0;i<_cdata.size();i++) {
        entries.add(new Entry(
            (float)_cdata.getDouble(i),i));
        xValList.add(new Integer(i).toString());
    }
}
```

Nach dem erfolgreichen Anlegen der beiden Datenfelder können wir die Informationen in ein für *MPAndroidChart* verständliches Format bringen und in das Steuerelement weiterleiten:

```
LineDataSet aDS=new LineDataSet( entries, "Testdaten");
_view.setData(new LineData(xValList, aDS));
}
```

In *render()* ist nun insofern Extraarbeit erforderlich, als wir im ersten Schritt ein Array mit Informationen bevölkern müssen. Dieses wird im nächsten Schritt in das *LineChart* eingebaut, um so im Rahmen der Diagramminitialisierung zur Verfügung zu stehen.

Damit müsste die Klasse eigentlich zur Anzeige von Daten befähigt sein: Ein im Setter platzierter Breakpoint bestätigt, dass die Engine die angelieferten Informationen ordnungsgemäß in das Steuerelement weiterleitet. Leider führt die Ausführung des Programms nach wie vor zu einem leeren weißen Bildschirm. Kenner von *MPAndroidChart* versuchen an dieser Stelle im ersten Schritt instinktiv, das Steuerelement unter Nutzung der *setLayoutParams*-Methode zur Ausbreitung zu animieren.

Leider führt diese Vorgehensweise nicht zum Ziel. React Native erledigt das Layouten anhand der CSS-Boxkomponente, die in der vorliegenden Version keine Rücksicht auf Werte wie *FILL_PARENT* nimmt. Der einfachste Weg zur Umgehung dieses Problems ist das Nutzen des *style*-Parameters, der bei Angabe von *height*, *flexDirection* und *row* für die Zuweisung von Platz sorgt:

```
render(){
    var values = [12, 14, 2, 7, 9, 5, 10];
    return (<NMGLineChart cdata={values}
        style={{flexDirection: 'row', height: 100,
        padding: 20}} />);
}
```

Damit ist die Arbeit an unserem hauseigenen Steuerelement abgeschlossen. Eventuell notwendige Ereignisse lassen sich

Listing 3: onCreate

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    mReactRootView = new ReactRootView(this);
    mReactInstanceManager =
        ReactInstanceManager.builder()
            .setApplication(getApplication())
            .setBundleAssetName("index.android.bundle")
            .setJSMainModuleName("index.android")
            .addPackage(new MainReactPackage())
            .setUseDeveloperSupport(BuildConfig.DEBUG)
            .setInitialLifecycleState(LifecycleState.RESUMED)
            .build();
    mReactRootView.startReactApplication(
        (mReactInstanceManager, "MyAwesomeApp", null));
    setContentView(mReactRootView);
}
```

unter Nutzung des in React Native integrierten Event-Subsystems für JavaScript exponieren. Weitere Informationen dazu finden sich unter <https://facebook.github.io/react-native/docs/native-components-android.html> im Abschnitt *Events*.

Komplettes Neudesign

Es steht außer Frage, dass ein komplettes Neudesign einer schon vorhandenen Applikation den idealen Weg zu einem React-basierten Projekt darstellt: Wenn jedes Java-Element von Hand portiert wird, ist das Resultat am saubersten.

Dank konsequenter Nutzung des Gradle-Buildsystems lässt sich diese Einbindung mit minimalem Aufwand realisieren. In der zur App gehörenden *build.gradle* muss eine Dependency auf *compile "com.facebook.react:react-native:+"* angelegt werden, während die Projekt-Build-Datei einen Verweis auf das Maven-Repository eingeschrieben bekommt.

Für die eigentliche Belebung wird eine Activity mit einer *ReactRootView* ausgestattet: Das Steuerelement verhält sich wie jedes andere Widget. Eine gute Beispielimplementierung würde *onCreate* wie in Listing 3 bevölkern.

Für perfektes Funktionieren muss die *RootView*-Instanz noch mit einigen Events versorgt werden. Der Implementierungsaufwand ist dann in etwa mit dem Deployment einer komplexeren Steuerelementbibliothek vergleichbar. ■



Tam Hanna

ist Autor, Trainer und Berater mit den Schwerpunkten Webentwicklung und Webtechnologien. Es lebt in der Slowakei und leitet dort die Firma Tamoggemon Holding k.s. Er bloggt sporadisch unter:

www.tamoggemon.com

SWIFT-PLAYGROUNDS-APP

Programmieren am iPad

Mit der neuen App Swift Playgrounds kann man direkt auf dem iPad programmieren.

Apples alljährliche Entwicklerkonferenz WWDC hält typischerweise für Entwickler eine Vielzahl von Neuerungen und Überraschungen bereit. Dabei gab es in diesem Jahr ein ganz besonderes Highlight für alle iPad-Besitzer: Zusammen mit iOS 10 – der neuen Version des Betriebssystems für iPhone, iPad und iPod touch – veröffentlicht Apple auch eine neue iPad-App namens Swift Playgrounds (Bild 1).

Wie der Name bereits andeutet, bringt Apple damit die bereits aus Xcode bekannten Playgrounds für iOS nun auch direkt auf das iPad. Das allein ist aber noch nicht alles. Darüber hinaus besitzt Swift Playgrounds mehrere verschiedene Aufgaben und Tutorials, die dem Nutzer die Programmierung im Allgemeinen sowie die Programmierung mit Swift im Speziellen erklären und näherbringen sollen. Apple platziert die App damit auch als klare Lernplattform für angehende Programmierer, um diesen die Grundlagen beizubringen und mit auf den weiteren Weg zu geben.

Swift Playgrounds wird kostenlos mit dem Release von iOS 10 zum Download im App Store zur Verfügung stehen. App-Entwickler, die die Beta von iOS 10 auf einem iPad installieren, oder Teilnehmer des iOS-Beta-Programms erhalten bereits jetzt Zugriff auf das in der iOS-10-Beta vorinstallierte und nicht löschbare Swift Playgrounds.

Wie beschrieben liegt einer der Schwerpunkte von Swift Playgrounds darin, die Grundlagen der Programmierung zu



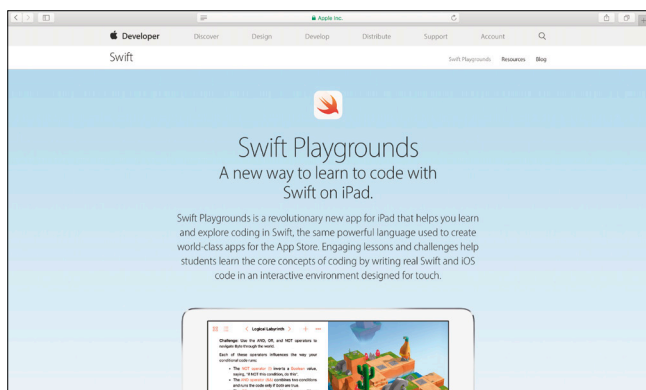
Foto: Shutterstock / Kit8.net

vermitteln. Aus diesem Grund gibt es innerhalb der App verschiedene Kurse, in denen die Software-Entwicklung in einzelne Schritte und Lessons unterteilt wird, die dann nach und nach abgearbeitet werden können. Der wohl bisher bekannteste dieser Kurse hört auf den Namen »Fundamentals of Swift« und wurde bereits von Apple selbst auf der diesjährigen WWDC kurz vorgestellt (Bild 2).

Dabei steuert man ein kleines Wesen namens Byte durch kurze Levelabschnitte, in denen man passende Code-Fragmente korrekt miteinander kombiniert oder eigene neue Befehle und Funktionen erstellt.

Die App führt dabei Schritt für Schritt verschiedene Konzepte der Programmierung ein. Geht es anfangs nur um das korrekte Aneinanderreihen korrekter Befehle, erstellt man kurze Zeit später eigene Funktionen und arbeitet mit Abfragen und Schleifen. Komplexe Zusammenhänge werden leicht und spielerisch erklärt und machen tatsächlich Spaß. Selbst erfahrene Entwickler können den gelungenen Levels und Aufgaben etwas abgewinnen und ihr Wissen festigen und vertiefen.

Erfolgreich abgeschlossene Level werden innerhalb des Kurses markiert. Über eine entsprechende Schaltfläche lassen sich alle Abschnitte und Lessons des Kurses aufrufen, und auf einen Blick ist zu erkennen, welche davon schon erfolgreich bearbeitet wurden und welche nicht (Bild 3). Damit ist auch ein schnelles Wechseln und Springen durch die Lessons



Mit Swift Playgrounds ist erstmals das Programmieren auf dem iPad möglich (Bild 1)

möglich, ohne dafür aufwendig über die beiden Pfeil-Schaltflächen neben dem Titel einzeln von Seite zu Seite des Playgrounds zu wechseln.

Ist man mit dem geschriebenen Code zufrieden, kann man diesen direkt auf dem iPad ausführen lassen. Im Fall des genannten Kurses »Fundamentals of Swift« stapft dann das Wesen namens Byte los und führt die Befehle aus, die man im eigenen Programm festgelegt hat. Ziel ist es dabei, Edelsteine einzusammeln und/oder Schalter umzulegen.

Das Schöne an diesem Konzept ist, dass man jeden einzelnen ausgeführten Befehl in einer schön animierten Live-Vorschau nachvollziehen kann. Wenn irgendetwas nicht passt oder ein wenig hakt, sieht man sehr schnell, an welcher Stelle etwas schief läuft, und kann entsprechend eingreifen.

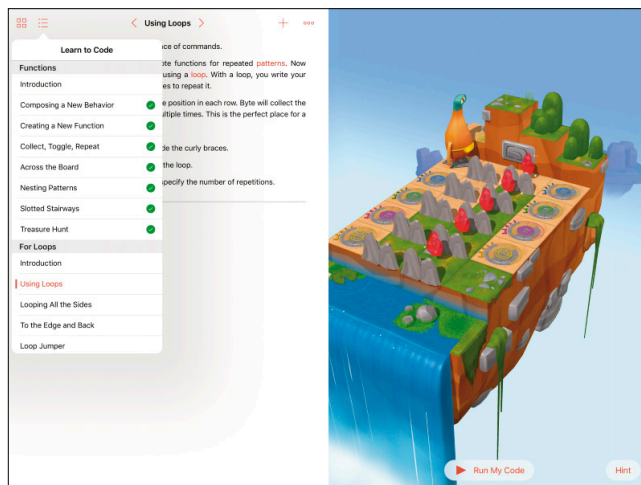
Viele vorgefertigte Funktionen

Viel Code muss in diesem Kurs übrigens noch nicht geschrieben werden. Die meiste Zeit werden bereits Funktionen vorgegeben und die Aufgabe des Nutzers besteht darin, diese sinnvoll und passend aneinanderzureihen.

Das mag einen nicht zu einem vollwertigen Programmierer machen, weckt aber das Verständnis für grundlegende Problematiken und Gedankengänge in der Software-Entwicklung, und genau das ist ja auch der von Apple angestrebte Sinn und Zweck der Sache.

Dennoch ist es problemlos möglich (und auch manchmal notwendig) innerhalb der Playgrounds in der App komplett eigenen Code zu schreiben; ohne Einschränkungen. Frameworks wie UIKit stehen vollumfänglich in Swift Playgrounds zur Verfügung und können somit mit allen Klassen und Funktionen im eigenen Playground genutzt werden.

Kurse wie der genannte »Fundamentals of Swift« können direkt aus der App Swift Playgrounds heraus geladen werden (Bild 4). Dort findet sich ein (sehr wahrscheinlich regelmäßig aktualisierter) Katalog an Lessons, die einfach ausgewählt und heruntergeladen werden können. Sie stehen dann in der Übersicht der eigenen Playgrounds zur Verfügung und können von dort aus aufgerufen und verwendet werden.



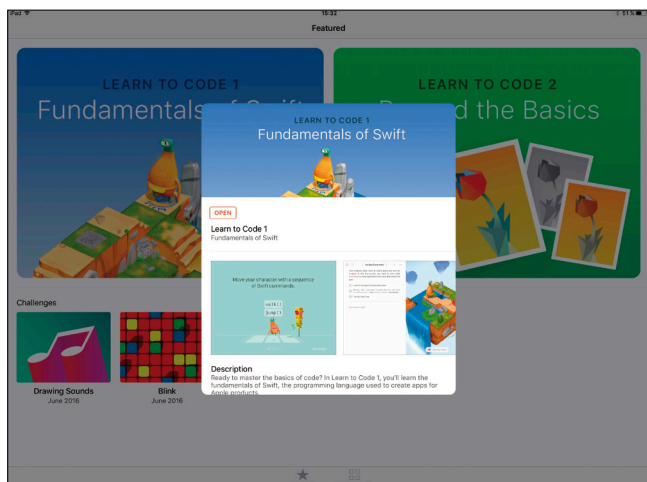
Die Schaltfläche oben links präsentiert die Inhalte eines Playgrounds mitsamt Fortschritt (Bild 3)

Damit das Programmieren auf dem iPad so flüssig wie nur möglich vonstatten gehen kann, hat Apple die integrierte Bildschirmtastatur für Swift Playgrounds ein wenig überarbeitet. Dabei hebt sich besonders eine Schnellauswahlleiste hervor, die am oberen Rand der Tastatur angezeigt wird und mögliche potenziell nächste Befehle enthält (Bild 5).

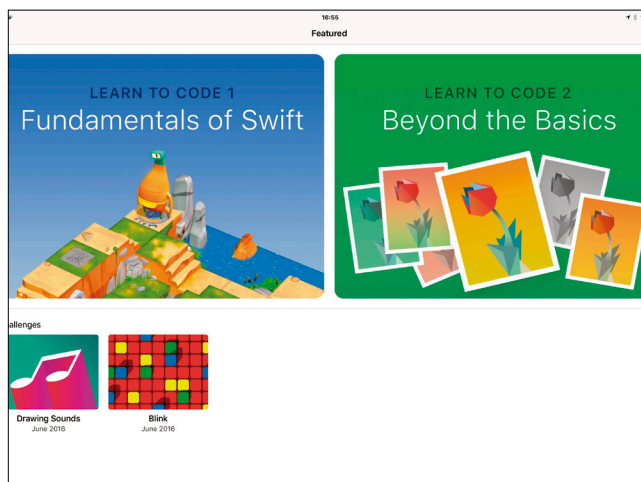
Es reicht ein Tap auf einen solchen Befehl, und umgehend wird er direkt in der passenden Form in den eigenen Code eingebunden.

Gerade bei den Lern-Lessons in den verschiedenen angebotenen Kursen der App ist diese Schnellauswahlleiste mitunter das einzige Eingabeelement, das man benötigt, um den zur Erfüllung der Aufgabe benötigten Code zu schreiben. Man wählt einfach nacheinander die gewünschten Befehle und Funktionen aus und sieht dann, was passiert.

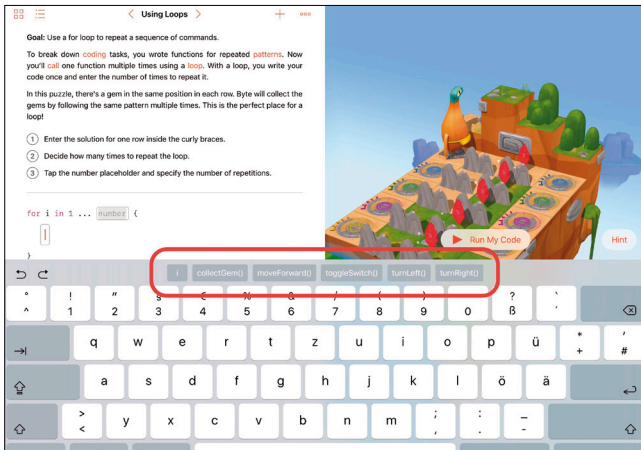
Diese Schnellauswahlleiste verändert sich dabei auch dynamisch und versucht, bestmöglich passende Befehle anzubieten. Dabei werden sogar im eigenen Code deklarierte Variablen oder Methoden mit einbezogen. ▶



Der Kurs »Fundamentals of Swift« erklärt die Grundlagen der Programmierung mit Swift (Bild 2)



Die verschiedenen zur Verfügung stehenden Kurse können direkt aus der App heraus geladen werden (Bild 4)



Über die Schnellauswahlleiste der Bildschirmtastatur lassen sich komfortabel neue Befehle erstellen (Bild 5)

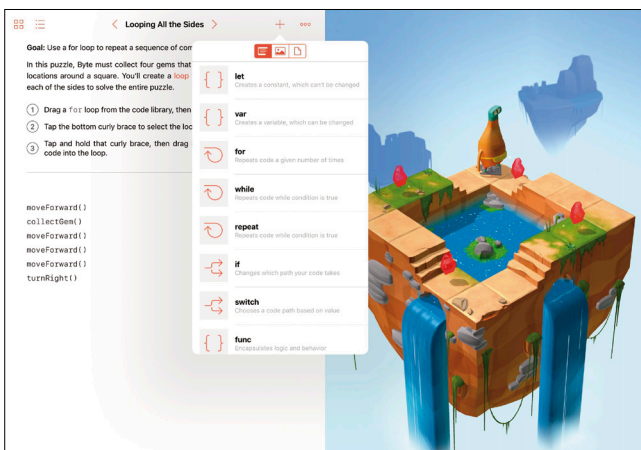
Darüber hinaus soll die finale Version von Swift Playgrounds auch über diverse Shortcuts bei den anderen Tasten der Tastatur verfügen, um beispielsweise schnell geschweifte Klammern zu erstellen. Leider stand diese Version der Tastatur aber in der zur Drucklegung getesteten Beta von Swift Playgrounds noch nicht zur Verfügung.

Es ist davon auszugehen, dass diese entweder in einer späteren Beta-Version von iOS oder mit dem Release der finalen Fassung von iOS 10 dann auch ihren Weg in die App finden wird.

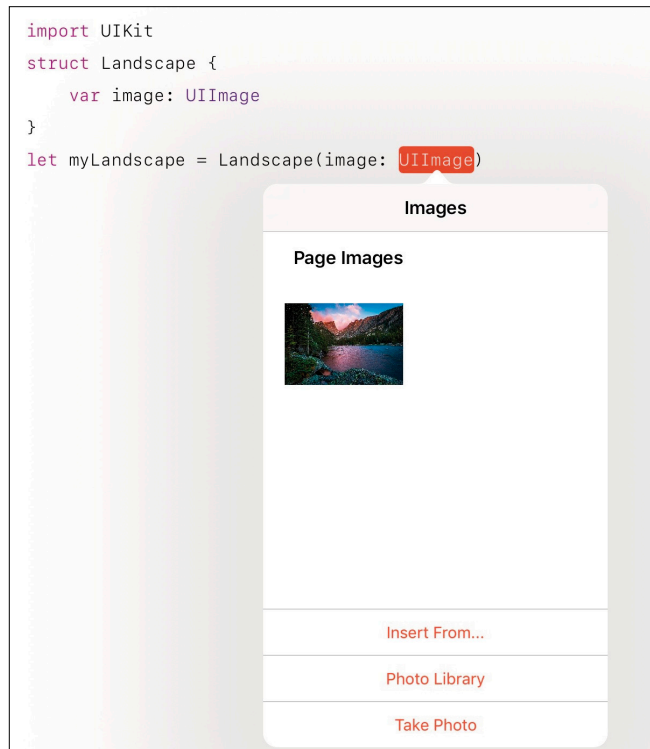
Code Snippets und Ressourcen

So schön, sinnvoll und angebracht die angepasste Bildschirmtastatur auch ist, so sind einige sich öfter wiederholende Abläufe damit immer noch recht mühsam zu erstellen (beispielsweise das Erzeugen von Funktionen oder Klassen). Daher bietet die App für viele dieser genannten Zwecke eine kleine Library mit Code Snippets an (Bild 6).

Ein einfacher Tap genügt, und es lassen sich neue Variablen und Konstanten, Schleifen, Abfragen, Funktionen, Protokolle, Enumerations, Structures und Klassen erstellen.



Über die Plus-Schaltfläche erlangt man Zugriff auf Code Snippets und sonstige Ressourcen des Playgrounds wie Bilder (Bild 6)



Bilder können schnell und einfach im Code des eigenen Projekts hinzugefügt werden (Bild 7)

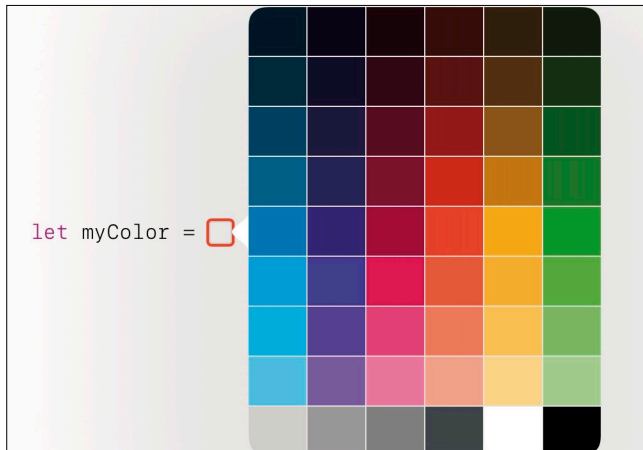
Zudem kann man über das kleine Segmented Control am oberen Rand dieses Pop-overs auch die Ressourcen eines Playgrounds wie beispielsweise Bilder einsehen. Darüber können auch gleichzeitig neue Ressourcen einem Playground hinzugefügt werden. Die Ressourcen selbst können auf Wunsch im Code des Playgrounds referenziert und verwendet werden. Das erweitert die Einsatzzwecke von Swift Playgrounds enorm und erlaubt es, auch Code mit Multimedia-Dateien auf dem iPad zu entwickeln und auszuführen.

Arbeit mit Zahlen, Bildern und Farben

Der Code-Editor von Swift Playgrounds verfügt aber noch über weitere Highlights. Zu diesen gehört beispielsweise das einfache Erstellen von Farben oder Bildern im eigenen Code. Apple bietet dafür in der Schnellauswahlleiste der Bildschirmtastatur eigene Schaltflächen an, mit denen einfach und schnell Bilder und Farben im Code erstellt und bearbeitet werden können.



Ein VorschauBild zeigt, welchen Wert ein bestimmtes Bild besitzt (Bild 8)



Swift Playgrounds bietet einen passenden Picker, um schnell eine Farbe zu setzen (Bild 9)

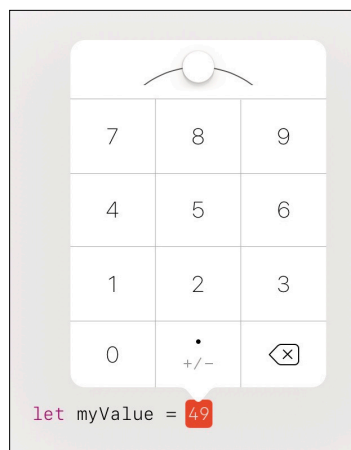


Über diese beiden Schaltflächen in der Schnellwahlleiste lassen sich neue Farben oder Bilder erstellen (Bild 10)

Wie alle Schaltflächen innerhalb der Schnellwahlleiste sind auch diese kontextsensitiv, sprich, sie tauchen nur dann auf, wenn es sinnvoll ist. Um einmal die genannte Funktionalität zu demonstrieren, habe ich einen neuen Playground erstellt und darin eine Structure namens *Landscape* definiert. Diese Structure verfügt lediglich über eine einzige Property namens *image* vom Typ *UIImage*, welche dazu dienen soll, tolle Landschaftsaufnahmen in einem Objekt dieses Typs festzuhalten. Durch diese Deklaration ist klar, dass innerhalb der Property *image* also ein Bild gespeichert werden soll. Der überschaubare Code für dieses Beispiel sieht so aus:

```
import UIKit
struct Landscape {
    var image: UIImage
}
```

Erstelle ich nun aufgrund dieser Basis ein neues Objekt vom Typ *Landscape* und rufe dabei den dazu passenden Initializer auf, wird direkt ein Pop-over für den *image*-Parameter angezeigt. Über diesen kann ich auf alle Bilder, die Teil meines Playgrounds sind, zugreifen oder ein Bild aus einer anderen Quelle meinem Playground hinzufügen (Bild 7). Anschließend reicht es aus, das gewünschte Bild für das Objekt auszuwählen; fertig ist das Erstellen eines *UIImage*-Objekts.



Ein eigenes Pop-over erleichtert die Eingabe von Zahlenwerten (Bild 11)

```
let firstName = "Thomas"
let lastName = "Sillmann"
let fullName = "\(firstName) \(lastName)"
```

Thomas Sillmann

Add viewer

Mittels **Quick Look** lassen sich die Werte von Variablen und Konstanten in grafischer Form anzeigen (Bild 12)

Anschließend lässt sich innerhalb des Playgrounds auch immer ein kleines Vorschaubildchen mit dem zugewiesenen Bild anzeigen. Somit ist auch beim Überfliegen des Codes schnell ersichtlich, welchen Wert ein bestimmtes Bild besitzt (Bild 8).

Äquivalent dazu verhält es sich mit Farben. Auch für Elemente vom Typ *UIColor* bietet Swift Playgrounds einen passenden Picker, um schnell eine Farbe zu setzen (Bild 9).

Die genannten Elemente für Bilder und Farben lassen sich auch nicht nur bei Objektzuweisungen verwenden. In der Schnellwahlleiste der Bildschirmtastatur stehen sie mit jeweils einem eigenen Befehl zur Verfügung, um entweder direkt eine Farbe oder ein Bild im eigenen Code des Playgrounds einzufügen (Bild 10).

Ach ja, und abschließend ist an dieser Stelle noch ein weiteres Pop-over zur Vereinfachung von Zahleneingaben zu nennen. Dazu reicht es einfach aus, auf einen Zahlenwert im Code des Playgrounds zu tippen, und Swift Playgrounds stellt eine eigens für Zahlen angepasste Maske zur Verfügung, um den Wert zu verändern (Bild 11).

Dabei kann der Slider am oberen Rand dieses Pop-overs dazu verwendet werden, schnell durch Wischen nach links den Wert zu verringern beziehungsweise durch Wischen nach rechts zu erhöhen. Dabei ändert sich das User-Interface-Pop-over zu einem vollständigen Kreis, durch den man dann einfach durch Wischen in die beiden möglichen Richtungen den Wert verringert oder erhöht. Das Ergebnis wird umgehend in den Code des Playgrounds übernommen und ist somit sofort sichtbar.

Quick Look

Überhaupt ist Swift Playgrounds – genau wie die mit Xcode erstellten Playgrounds auf dem Mac – Quick-Look-fähig. Das bedeutet, dass Sie für die verschiedensten Werte Ihrer Variablen und Konstanten direkt eine Vorschau anzeigen lassen können; bei Strings sehen Sie somit den Text der jeweiligen Variable oder Konstante, bei Bildern das zugehörige Bild und bei Graphen den entsprechenden Verlauf (Bild 12).

Um den Wert einer Variablen oder Konstanten im wahrsten Sinne des Wortes zu sehen, reicht es aus, auf die Vorschau-schaltfläche rechts vom jeweiligen Element zu tippen. Zu beachten ist aber dabei, dass diese Vorschau erst dann auftaucht, wenn der Code des Playgrounds einmal kompiliert wurde. Diese Kompilierung können ►

Sie über die Schaltfläche **Run My Code** starten und ausführen lassen (Bild 13). Läuft diese erfolgreich durch, erscheinen auch die zugehörigen Quick-Look-Schaltflächen an den passenden Stellen.

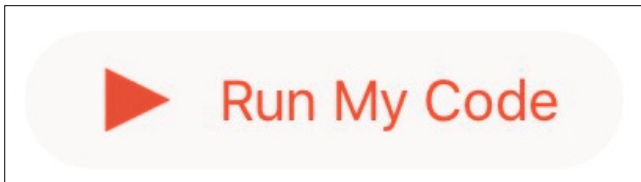
Wenn Sie Ihren Code ändern, müssen Sie diesen aber jedes Mal neu über die Schaltfläche *Run My Code* kompilieren; eine automatische Ausführung des Codes wie in den Playgrounds in Xcode unterstützt Swift Playgrounds nicht. Beachtet man das nicht, kann das zu Irritationen führen, wenn die Quick-Look-Vorschau nach einer Code-Änderung einen anderen Wert als erwartet anzeigt.

Vorlagen für neue Playgrounds

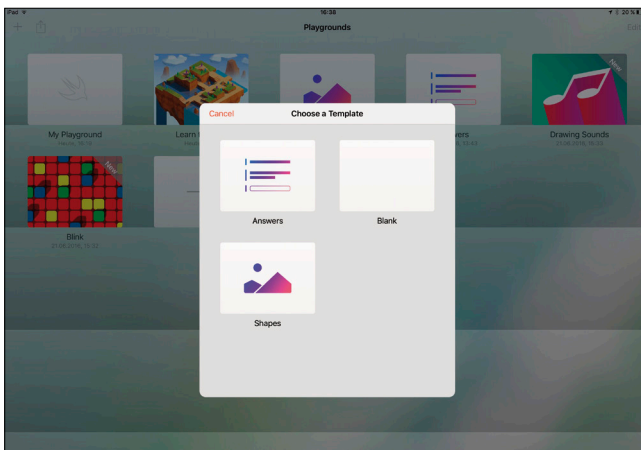
Das Erlernen von Swift mit Hilfe der verschiedenen vorgefertigten Kurse und die mannigfaltigen Möglichkeiten zum Schreiben von Code sind die eine Attraktion von Swift Playgrounds. Daneben begeistert aber auch die Möglichkeit, komplett autarke und unabhängige Playgrounds zu erstellen, so, wie man sie bereits von Xcode her kennt.

Dazu tippt man in der Übersicht der Playgrounds entweder auf die Plus-Schaltfläche in der oberen linken Ecke und anschließend auf *Create Playground*, oder man tippt direkt auf die große Plus-Schaltfläche am Ende inmitten der Übersicht der verwendeten Playgrounds. Anschließend öffnet sich eine neue Ansicht, die das Erstellen eines neuen Playgrounds ermöglicht.

Die App bringt dazu drei verschiedene Vorlagen mit, um einen passenden Einstieg für die Entwicklung eines eigenen Playgrounds auszuwählen (Bild 14).



Über die Schaltfläche **Run My Code** führen Sie den Code eines Playgrounds aus (Bild 13)



Beim Erstellen eines neuen Playgrounds stehen drei verschiedene Vorlagen zur Wahl (Bild 14)

Über diese Playgrounds lassen sich die vorgestellten Features zur Bedienung und zum Schreiben von Code (wie beispielsweise das Erstellen von Bildern und Farben mittels dazugehöriger passender Picker) ideal anwenden. Überhaupt hat man in diesen eigens erstellten Playgrounds komplett freie Hand und kann alles erstellen, wonach einem ist. Die Lernkurse hingegen sind eben darauf ausgelegt, konkrete Problemstellungen möglichst leicht und übersichtlich zu lösen.

Die einfachste Form, um einen neuen Playground zu erstellen, ist, die *Blank*-Vorlage zu wählen. Dann erhält man nämlich exakt das, was auch versprochen wird: eine leere hellgraue Fläche, die darauf wartet, mit Code befüllt zu werden. Sie können direkt darauf losschreiben und alle Funktionen der Swift Standard Library zur Programmierung nutzen. Wenn Sie weitere Frameworks wie das UIKit-Framework benötigen, so importieren Sie diese einfach zu Beginn des Playgrounds (eben genauso, wie es auch in Xcode-Projekten gehandhabt wird).

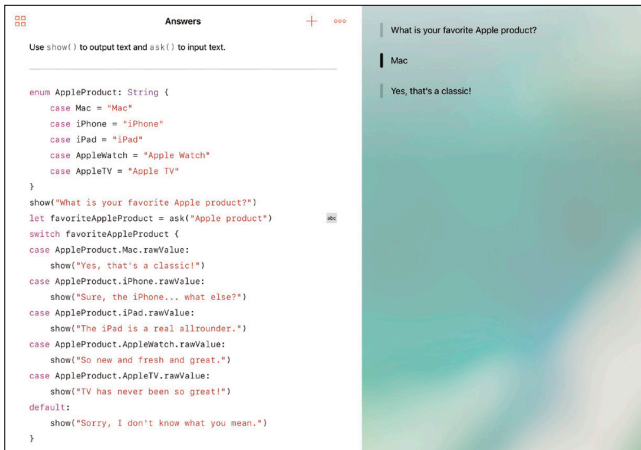
Wer es etwas spezifischer mag, findet in Swift Playgrounds zwei weitere Vorlagen, um angepasste Playgrounds zu erstellen. Die erste davon ist *Answers*. Wie der Name bereits andeutet, können damit Projekte umgesetzt werden, die Nutzerangaben abfragen und anschließend auswerten. Auf dem Mac ist das typischerweise das Metier von einfachen Konsolenanwendungen, die Eingaben entgegennehmen und anschließend entsprechende Aktionen ausführen.

Damit das Ganze funktioniert, bringt die *Answers*-Vorlage direkt zwei voll funktionsfähige Methoden mit: *show(_)* und

Listing 1: User-Abfrage

```
enum AppleProduct: String {
    case Mac = "Mac"
    case iPhone = "iPhone"
    case iPad = "iPad"
    case AppleWatch = "Apple Watch"
    case AppleTV = "Apple TV"
}

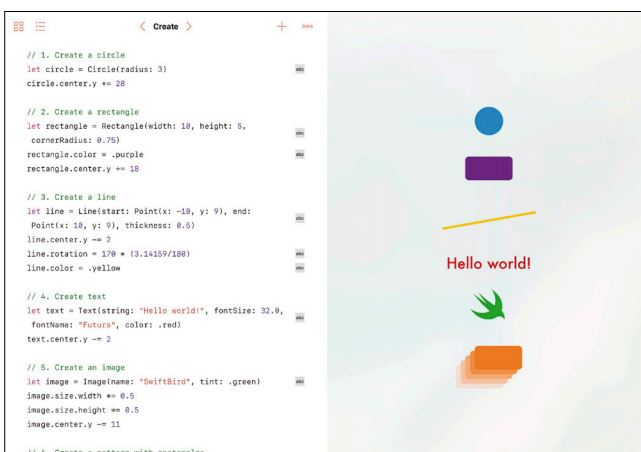
show("What is your favorite Apple product?")
let favoriteAppleProduct = ask("Apple product")
switch favoriteAppleProduct {
case AppleProduct.Mac.rawValue:
    show("Yes, that's a classic!")
case AppleProduct.iPhone.rawValue:
    show("Sure, the iPhone... what else?")
case AppleProduct.iPad.rawValue:
    show("The iPad is a real allrounder.")
case AppleProduct.AppleWatch.rawValue:
    show("So new and fresh and great.")
case AppleProduct.AppleTV.rawValue:
    show("TV has never been so great!")
default:
    show("Sorry, I don't know what you mean.")
}
```

Mit der **Answers-Vorlage** können Nutzereingaben einfach in einem Playground ausgewertet werden (Bild 15)

`ask(_)`. Beide nehmen als Parameter einen String entgegen. `show(_)` gibt den übergebenen String in einer separaten Ansicht rechts vom Code aus und dient dazu, beispielsweise Fragen zu formulieren. `ask(_)` hingegen blendet in dieser rechten Ansicht ein Textfeld ein und zeigt den der Methode übergebenen String als Platzhalter für eben jenes Textfeld an. Der Nutzer kann dieses Textfeld auswählen und anschließend eine Eingabe tätigen. Diese Eingabe wird dann entweder per Return-Taste oder durch Tippen auf eine *Submit*-Schaltfläche bestätigt. Im Playground wird der eingegebene Wert von der `ask(_)`-Methode zurückgegeben und kann so beispielsweise einer Variablen oder Konstanten zugewiesen werden, um nach der Eingabe eine Auswertung vorzunehmen.

Listing 1 zeigt ein Beispiel, wie ein solcher Code für einen Answers-Playground aussehen kann. Dabei wird in einer Enumeration eine Liste von Apple-Produkten erstellt, aus denen der Nutzer seinen Favoriten wählen soll. Ihm wird eine entsprechende Frage gestellt, und seine Eingabe wird anschließend mittels einer *switch*-Abfrage ausgewertet. Es folgt eine passende Rückmeldung an den Nutzer, abhängig davon, welche Eingabe er getätigt hat (Bild 15).



Mit der **Shapes-Vorlage** lassen sich schnell und einfach Formen im eigenen Code umsetzen (Bild 16)

Mit der letzten Vorlage – *Shapes* – ist es möglich, schnell grafische Formen zu erstellen (Bild 16). Dabei verfügen mit dieser Vorlage erstellte Playgrounds direkt über mehrere Seiten, die den grundlegenden Code bereitstellen und durch die mittels der beiden Pfeil-Schaltflächen links und rechts vom Titel durchgewechselt werden kann. Dort wird auch erklärt, wie man mit den bereitgestellten Typen und Funktionen arbeitet, um so beispielsweise auch eine Touch-Erkennung umzusetzen und Objekte auf dem Display bewegen zu können. Auf einer zusätzlichen Seite des Playgrounds namens *Canvas* findet dann der eigens geschriebene Code Platz, der typischerweise die zur Verfügung stehenden Formen und Medien ausnutzen kann.

Fazit

Mit Swift Playgrounds gibt es ab iOS 10 die Möglichkeit, direkt auf dem iPad zu programmieren; zumindest in Swift. Aufgrund der Möglichkeit, aus der iOS-Entwicklung bekannte Frameworks wie UIKit auch in Swift Playgrounds auf dem iPad zu importieren, steht Entwicklern eine gigantische Fülle an APIs zur Verfügung, um auch unterwegs auf Apples Tablet zu programmieren.

Daneben liegt der Schwerpunkt der App klar auf dem Fokus, das Programmieren überhaupt zu erlernen und die eigenen Fähigkeiten auszubauen. Die ersten bereits in der Beta zur Verfügung stehenden Kurse sind dabei bereits großartig und verspielt umgesetzt und bereiten tatsächlich jede Menge Spaß. Wir dürfen gespannt sein, was uns hier in Zukunft noch erwarten wird und wie die Auswahl zum finalen Release der App im Herbst zusammen mit iOS 10 aussehen wird.

Apple hat es verstanden, die Bedürfnisse von Entwicklern vom Desktop-Rechner auf das iPad zu übertragen. Die optimierte Bildschirmstastatur erlaubt auch dann flottes Programmieren, wenn keine externe Bluetooth- oder Smart-Connector-Tastatur angebunden ist. Einige der auf der WWDC vorgestellten Funktionen fehlten zwar noch in der getesteten Beta-Version, bis zum Release sollten diese aber vollständig implementiert sein.

Wer Swift Playgrounds einmal selbst testen möchte, benötigt ein iPad sowie eine Mitgliedschaft im Apple Developer Program. Darüber kann dann die Beta von iOS 10 bezogen und auf dem iPad installiert werden (achten Sie lediglich darauf, dass Ihr iPad auch tatsächlich iOS 10 noch unterstützt, eine Übersicht der unterstützten Geräte finden Sie auf der Apple-Website). Swift Playgrounds steht dort dann als vorinstallierte Standard-App zur Verfügung. ■



Thomas Sillmann

ist iOS-App-Entwickler, Trainer und Autor. Freiberuflich tätig programmiert er für den App Store eigene Apps sowie Apps in Form von Kundenaufträgen. Er ist Autor eines erfolgreichen Fachbuchs und mehrerer Artikel in Fachzeitschriften.
www.thomassillmann.de

VERSION 3 DER GRAPHDATENBANK NEO4J

Entscheidende Features

Die aktuelle Version 3 der Graphdatenbank Neo4j bietet zahlreiche Neuerungen.

Seit einigen Jahren verbreiten sich Graphdatenbanken zunehmend über verschiedene (teilweise spezielle) Anwendungsgebiete in immer mehr Unternehmen. Deren Anwender erkennen, dass sich die Graphtechnologie insbesondere für Einsatzzwecke eignet, in denen Beziehungen eine wichtige Rolle einnehmen.

Vor allem für Massendaten mit komplexen Beziehungen gibt es eine große Anzahl unterschiedlicher Anwendungsgebiete mit steigender Tendenz. Daher schätzen viele Analysten der Informationstechnologie wie Forrester Research oder Gartner den Verbreitungsgrad von Graphdatenbanken für die kommenden Jahre in Unternehmen auf über 25 Prozent ein. Als Hersteller des am häufigsten eingesetzten Systems profitiert Neo Technology von diesen steigenden Wachstumsraten.

Vorteilhaft für die Weiterentwicklung von Neo4j erwies sich, dass viele Unternehmen der Forbes-Global-2000-Liste wie Accenture, Cisco, Deutsche Telekom, Telenor und Walmart zu den Kunden des Datenbankherstellers gehören. Mit ihnen sammelte Neo Technology über größere Anwendungssysteme wichtige Erkenntnisse zu den real anzutreffenden Engpässen.

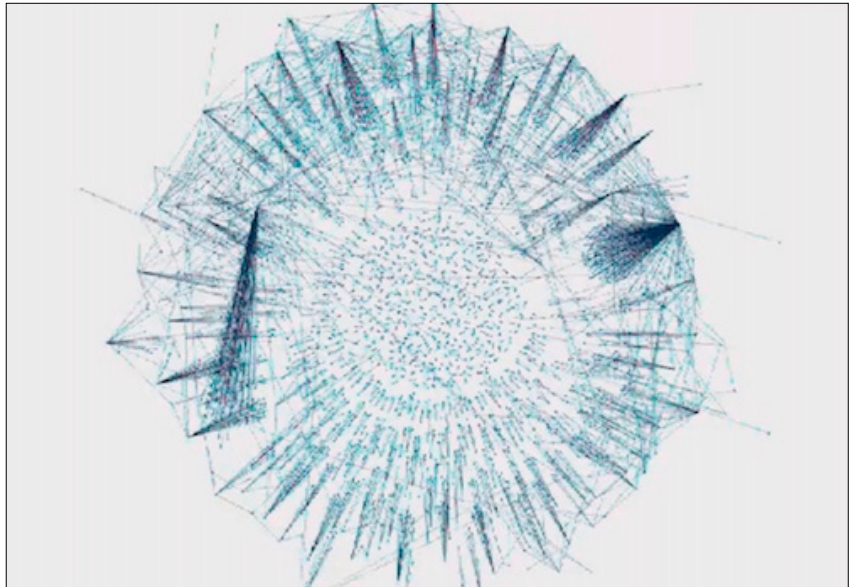
Auf Basis dieser vielseitigen Erfahrungen implementierte der Hersteller für die aktuelle Version 3 von Neo4j entscheidende neue Features und Verbesserungen.

Ihre Tauglichkeit für die Praxis erwies die neue Version kürzlich beim International Consortium of Investigative Journalists (ICIJ) als Analyse-Tool einer sehr großen Datenbank. In der Öffentlichkeit erhielt das ICIJ-Projekt, besser bekannt unter dem Namen Panama Papers, erhöhte Aufmerksamkeit.

Komplette Überarbeitung der Neo4j-internen Speicherstrukturen

Mit Version 3 verfolgte Neo Technology das Ziel, vorhandene Systemgrenzen – vor allem die Größe der Graphen betreffend – zu beseitigen. So entschloss sich der Hersteller, für das Herzstück der Systemarchitektur, den Datenspeicher (Data Store, manchmal auch Storage Layer genannt), ein vollständiges Reengineering durchzuführen.

Auf der Basis dieses Redesigns unterstützt Neo4j in der neuen Version erstmals Giant Graphs, also sehr große Graphen mit massenhaft vielen Knoten und Kanten. Diese Giant Graphs (gigantische Graphen) (Bild 1) besitzen mehr als das



Sehr große Graphen mit Tausenden und riesige Graphen mit Milliarden von Knoten stellen eine Herausforderung für Graphentheoretiker dar (Bild 1)

Tausendfache an Elementen (Knoten und Kanten) als alle bisher in Neo4j maximal realisierbaren Graphen.

Bereits der Erfinder des Webs Sir Tim Berners-Lee prägte für das semantische Web den Begriff des Giant Global Graph. Neuere Erkenntnisse der Graphentheorie zeigen, dass die Modellierung und Verarbeitung von Netzen eine dynamische Generierung des Graphen notwendig macht. Erst eine Dynamisierung erlaubt die Analyse von Graphen, die einer zeitlichen Entwicklung unterliegen, sich also über die Zeit hinweg ändern.

Simulation und Prognosen

Damit unterstützt der Graph so wichtige Prozesse wie Simulation, Prognosen oder Wachstum; gleichzeitig ermöglicht Data Mining in der Graphdatenbank es, Gesetzmäßigkeiten zu erkennen. Und schließlich macht eine Dynamisierung die Graphtechnologie für das maschinelle Lernen zugänglich.

Die neue Storage Engine von Neo4j beseitigt vorhandene Grenzen für Graphen mit Datensätzen der Vorgängerversion, die im zweistelligen Milliardenbereich lagen (34,4 Milliarden Knoten). Für die Enterprise Edition der Version 3 existiert diese Grenze nicht mehr, wie Kunden durch die Zusammenarbeit mit dem Hersteller bestätigen, deren Graphen mehrere Milliarden an Daten umfassen.

Gleichzeitig erlaubt die überarbeitete Storage Engine der Enterprise Edition eine höhere Skalierbarkeit des Gesamtsystems bei gleichbleibend hoher Performance. Die bisherigen

MessagePack – ein kompaktes Austauschformat

Als binäres, serialisiertes Format beschreibt MessagePack gängige strukturierte Daten (angelehnt an das JSON-Format) für einen einfachen und kompakten Austausch.

Implementierungen gibt es in vielen verschiedenen Sprachen: C, C++, Go, Java, JavaScript, PHP, Python, Ruby, Scala und weitere. Generell tauscht MessagePack analog zu JSON Daten zwischen vielen gängigen Programmiersprachen aus, ist dabei aber schneller und kompakter.

Schranken eliminiert eine neuartige Technik, die Dynamic Pointer Compression. Mit ihr erweitert Neo4j den Adressraum des Graphen, falls notwendig. Verbessert wurde auch die bereits seit Längerem vorhandene komprimierte Speicherung der Daten.

Damit unterstützt das Datenbank-Management-System (DBMS) für seine Graphen eine unbeschränkte Anzahl von Knoten, Kanten und Properties. Für gigantische Graphen fiel zudem die Grenze der Indexstrukturen, sodass in Version 3 erstmals Indizes einen unbeschränkt großen Umfang einnehmen können.

Im Zusammenhang mit der Datenhaltung implementiert Neo Technology einen intelligenten Algorithmus, manchmal auch Smart Algorithmus genannt, der den gesamten Speicherbereich optimiert. Damit nutzt Neo4j den verfügbaren Speicher extrem effizient für alle Elemente der Graphdatenbank. Letztendlich erfolgte die Umsetzung dieser Features unter der Prämisse, weiterhin für den Graphen eine native Verarbeitung (auch Index-free Adjacency genannt) über seine Kanten sicherzustellen.

Kontinuierliche Verbesserungen für das Zugriffsverhalten auf den Graphen

Alleinige Beseitigung von Grenzen durch Neuerungen in der Systemarchitektur macht Graphdatenbanken noch lange nicht für eine Verarbeitung von Massendaten zugänglich. Leider geben auch externe Interfaces wie Dateisysteme oder die Geschwindigkeiten von Festplatten Begrenzungen vor, mit denen das DBMS gezielt umgehen muss.

Einen wesentlichen Fortschritt für die Verarbeitung der Zugriffe brachte die Nutzung von Lucene durch direkte Integration in Neo4j: Verzichtete man in den Anfangszeiten noch vollständig auf Indexstrukturen, so kam doch später zunehmend in den Anwendungen Bedarf für schnellere Datenzugriffe auf. So gelang es Spezialisten durch Hacking des Index-Managers von Neo4j, sogenannte manuelle Indizes aufzubauen.

Aufgrund steigenden Bedarfs nach dem manuellen Indextyp unterstützte der Hersteller diesen über Cypher mit der *START*-Anweisung. Um den damit verbundenen Zusatzaufwand zu minimieren, führte Neo4j Version 1.4 sogenannte automatische Indizes ein. Die Definition der automatischen Indizierung nahm man in der *neo4j.properties*-Konfigura-

tionsdatei vor. Den größten Fortschritt für das DBMS brachte die Version 2 mit der Einführung von Indexstrukturen auf der Schemaebene; zur Differenzierung gegenüber den anderen Indextypen bezeichnet man diese als Schema-Indizes. Erstmals erhält ein Anwendungsprogramm wirkliche Unabhängigkeit vom implementierten Zugriffsmechanismus – das DBMS selbst entscheidet über den Zugriffsweg.

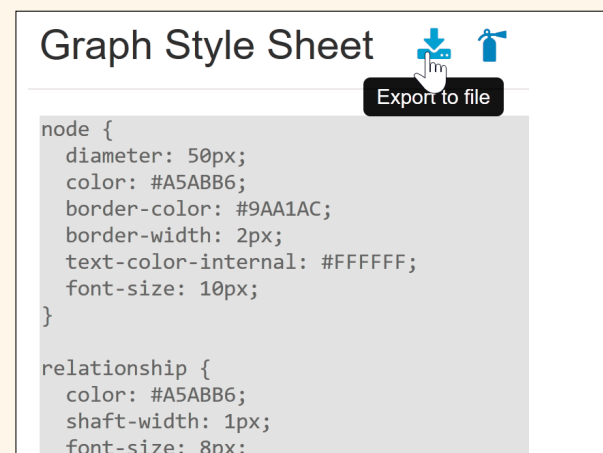
Analog zu SQL benötigt auch Cypher seitdem keine Vorgabe mehr, ob und welcher Index für den Zugriff auf die Daten zu verwenden ist. Cypher zeichnet sich jetzt zu Recht als wirklich deklarative Sprache aus.

Erst diese Unabhängigkeit erlaubt den Einsatz eines Optimierers, um abhängig von verschiedenen Parametern zu entscheiden, welcher Zugriffsweg als der optimale unter den aktuellen Umständen zu bewerten ist. Mit Version 2.2 gelang es Neo Technology, einen neuen, sogenannten kostenbasierenden Optimierer für Cypher einzuführen. Dieser ergänzt die bisherigen Regeln für Zugriffe um aktuelle Statistiken ►

Anschauliches Styling mit GRASS

Neo4j Browser Sync verwaltet auch GRASS (Graph Style Sheets) für die fortgeschrittene Visualisierung der Graphdatenbank.

Der Neo4j Browser verwendet GRASS, um die Elemente des Graphen (Knoten, Kanten) am Bildschirm anzuzeigen. GRASS basiert auf CSS (Cascading Stylesheets).



Das Dialogfenster Graph Style Sheet importiert GRASS-Dateien, exportiert den GRASS der aktuell angezeigten Graphdatenbank oder setzt die Einstellungen auf die Standardwerte zurück

Alle aktuell im visualisierten Graphen vorgenommenen Änderungen zu den Farben, Schriftarten, Größen, Titel oder Umrissen/Konturen seiner Elemente übernimmt der Neo4j Browser automatisch in das GRASS. Klickt man in der linken Tool-Leiste des Neo4j Browsers bei den Favorites (Symbol eines kleinen Sterns) auf die Schaltfläche *Graph Style Sheet*, so öffnet sich das gleichnamige Dialogfenster.

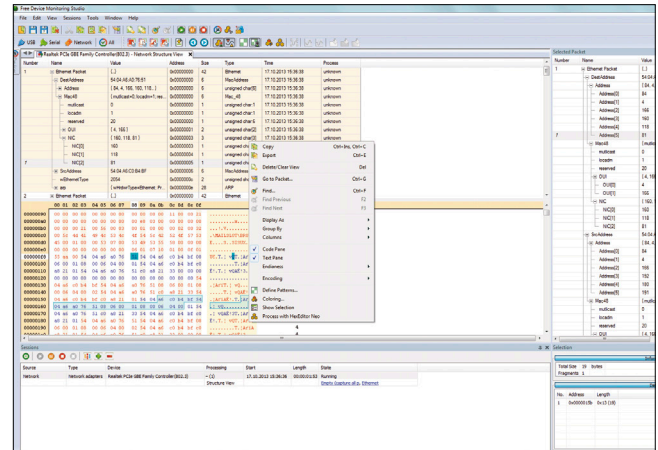
Lucene für mehr Performance und Skalierbarkeit

Lucene von der Apache Software Foundation stellt eine Programmierbibliothek zur Volltextsuche dar.

Auf Basis eines Index aus komprimierten Dateien liefert Lucene Suchergebnisse mit Ranglisten anhand verschiedener Suchalgorithmen. Als Technologie lässt sich Lucene für den Aufbau und Betrieb von Rechnerverbünden nutzen, um Hochverfügbarkeit und Lastverteilung zu unterstützen.

Um die Rechnerlast zu verteilen und dem Gesamtausfall des Systems entgegenzuwirken, partitioniert man einen Index horizontal über mehrere Server mittels sogenannter Shards.

Gruppen von Rechnern (Nodes) lassen sich als Cluster definieren, auf denen die Suchmaschine verteilt läuft. Dabei übernimmt einer der Nodes die Koordination als Master.



Binäre Netzwerkprotokolle erfordern den Einsatz eines Network Analyzers, um Ursachen eventueller Übertragungsprobleme zu erkennen und zu analysieren (Bild 2)

über die Ausprägung der Daten im Graphen.

Einheitliche Query-Optimierung und Aktualisierung von Lucene

Die neue Version 3 beseitigt die bisherige Beschränkung der Executionpläne auf reine Lesevorgänge und erweitert die kostenbasierte Zugriffsoptimierung auch auf Schreiboperationen.

Damit verwendet die Cypher-Engine erstmals für alle Operationen, ob Lesen oder Schreiben auf beziehungsweise von der Datenbank, eine einzige zentrale Optimierung. Gleichzeitig erreichte man eine bessere Performance bei der Ausführung von Cypher-Statements, einerseits durch Überarbeitungen bei der Ableitung der Executionpläne, andererseits durch den schlichten Wegfall des früher zusätzlich durch den Entscheidungsprozess erforderlichen Ausführungscodes.

Insgesamt brachte die Parallelisierung des Indizierungsprozesses bei der Anlage, Abfrage und Pflege von Indexstrukturen erhebliche Verbesserungen für das Laufzeitverhalten.

Daraus resultieren akzeptable Antwortzeiten für Abfragen von Zeichenketten mittels der Operatoren *ENDS WITH* oder *CONTAINS*, die Indexstrukturen einsetzen. Besonders verbesserte sich die Ausführung von *JOIN*-ähnlichen Operationen, die nicht auf Kanten beziehungsweise Beziehungen, sondern auf Datenwerten basieren. Neo4j kann temporäre Indexstrukturen im Hintergrund aufbauen, pflegen und sie oder ihre Statistiken für die Ableitungen kostenbasierter Zugriffswege nutzen.

Vorteilhaft erwies sich auch die Umstellung der seitens Neo4j eingesetzten Lucene-Implementierung von Version 3.6.2 auf 5.4.0; diese senkt den Speicherbedarf und auch die Zugriffszeit auf Indexstrukturen.

Der Einsatz einer neueren Lucene-Version macht die mit ihr verknüpften Vorteile unmittelbar den Funktionen des DBMS zugänglich; allerdings müssen bisher angelegte Graphen auf diese neue Version migriert werden. Damit verbunden ist ein höherer Zeitaufwand für die Umstellung vorhan-

dener Indizes älterer Graphdatenbanken auf die neue Neo4j-Version 3. Zusätzlich implementierte Neo Technology in seiner Lucene-Integration automatisches Sharding, um so die Grenze der Lucene-Indizes zu beseitigen.

BOLT für blitzschnelle Zugriffe auf den Graphen

Neo4j Version 3 nutzt im Unterschied zu sämtlichen Vorgängerversionen ein neu entwickeltes binäres Netzwerkprotokoll namens BOLT für die Kommunikation mit dem Graphen, um hochperformante Zugriffe auf die Datenbank auszuführen. Damit gewährleistet das DBMS zwar wie bisher eine einheitliche Zugriffstechnik auf seine Datenbestände, allerdings minimiert sich der gesamte Overhead bei der Kommunikation, sodass Zugriffe viel effizienter und zeitsparender als früher ausgeführt werden. Zudem erschließt Neo Technology auf Basis von BOLT für die Programmierung von Graphdaten-

Kronecker-Graphen zur Modellierung von Netzen

Das Kronecker-Produkt von Matrizen eignet sich für das rekursive Erzeugen/Konstruieren sogenannter selbstähnlicher Graphen.

Für maschinelles Lernen benötigt man Graphen mit zusätzlichen Eigenschaften, die man mathematisch analysieren und systematisch untersuchen kann. Sogenannte Kronecker-Graphen, die man mit dem Kronecker-Produkt erzeugt, besitzen derartige gemeinsame Netzwerk-Eigenschaften. Kronecker-Graphen modellieren effektiv die Struktur real existierender Netze.

Ferner gibt es einen schnellen und skalierbaren Algorithmus, KronFit genannt, der zum Modell des Kronecker-Graphen passt. Anstatt dynamische Graphen großen Umfangs in exponentieller Zeit zu analysieren, ermöglicht KronFit die Bearbeitung derartiger Graphen in linearer Zeit. Gleichzeitig findet man mit KronFit Parameter, die es erlauben, die Eigenschaften derartig großer gigantischer Graphen zu imitieren.

BOLT als mehrfach vorhandene Produktbezeichnung

Die neue Neo4j-Technologie BOLT hat mit den nachfolgenden gleichnamigen Produkten nichts gemeinsam und sollte nicht mit ihnen verwechselt werden:

- BOLT – eine an BASIC, C++, Java und Python orientierte Sprache für Programmieranfänger,
- BOLT – eine von AMD entwickelte C++-Template-Library,
- BOLT – die Peer-to-Peer (P2P) Networking Engine von Photon für die Spiel-Engine Unity.

banksystemen erstmals außer Java offiziell weitere Sprachen. Diese direkte Unterstützung für die Programmiersprachen C#.NET, JavaScript und Python durch den Hersteller bietet Anwendern wesentlich mehr Sicherheit für ihre in Neo4j getätigten Investitionen.

Mit dem neuen BOLT-Protokoll entfällt der bisherige Programmieraufwand für Entwickler beim Einsatz von REST durch die HTTP-Zugriffe – allerdings findet die Interaktion nun über ein derzeit noch proprietäres Binärformat statt (Bild 2). Gegenüber den vom Mensch lesbaren Textformaten (wie JSON oder XML) erlauben die BOLT-Zugriffe im Binärformat ein schnelleres Lesen der Daten aus Neo4j und damit verbunden höhere Streamingraten über das Netz. Die bessere Performance von BOLT resultiert auch aus dem wesentlich kompakteren Datenformat, das auf PackStream, einer von Neo Technology entwickelten Variante von MessagePack, basiert.

Entwickler formulieren CQL-Befehle (Cypher Query Language) und programmieren die Zugriffe auf die Graphdatenbank in einer der offiziell unterstützten Schnittstellen. Zur Ausführung der Datenbankzugriffe generiert das DBMS die erforderliche BOLT-Kommunikation und führt diese wesentlich schneller als in der Vorgängerversion aus. Zudem wird Verschlüsselung in der Transportschicht und Authentifizierung gewährleistet: BOLT verschlüsselt die Daten über TLP (Transport Layer Security; früher: Secure Sockets Layer (SSL) genannt).

Innerhalb und zwischen den BOLT-Modulen existieren direkte und eindeutige Zuständigkeiten gemäß den Prinzipien Separation of Concerns und Single Responsibility. Diese Realisierung gewährleistet für Neo Technology nicht nur eine hohe Modularisierung und damit verbunden leichtere Wartbarkeit, sondern bietet für die Programmierer auch konsistente Zugriffe auf die Graphdatenbank, was die einheitlichen Neo4j-Driver für die verschiedenen Programmiersprachen auf der Basis von BOLT unmittelbar belegen.

BOLT als Ausgangsbasis für weitere zukünftige Schnittstellen

Derzeit erklärt Neo Technology weder das Embedded Java-API noch das auf HTTP basierende REST als veraltet (im Java-Sprachgebrauch: deprecated). Vielmehr kann man diese Schnittstellen weiterhin in der Programmierung verwenden, allerdings sollte man neue Datenbanksysteme mit dem aktu-

ellen Neo4j-Driver der jeweiligen Sprache entwickeln. Nicht weil BOLT darauf abzielt, die REST-Kommunikation abzulösen, sondern wegen der mit dem Protokoll verbundenen ausgezeichneten Performance. Je größer die seitens Neo4j zu verwaltenden Graphen, desto deutlicher zeigt sich der Vorteil von BOLT innerhalb einer Anwendung. Als versioniertes Protokoll unterstützt ein Server gleichzeitig mehrere Versionen am Client.

Höherer Datendurchsatz

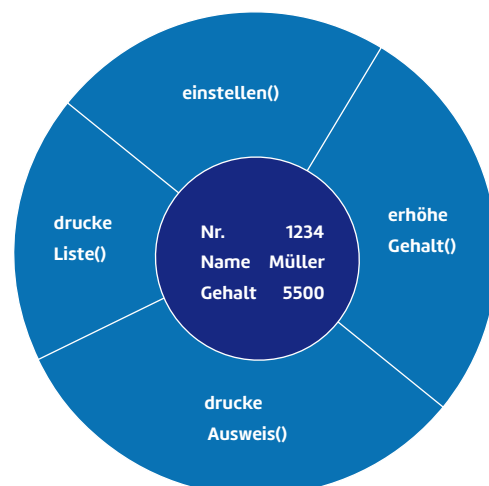
Es hat sich gezeigt, dass man mit BOLT über TCP oder WebSockets einen höheren Durchsatz mit einer geringeren Latenz erreicht. Ergänzend bieten die von Neo Technology bereitgestellten Neo4j-Driver einer bestimmten Programmiersprache erstmals eine Kapselung im Sinne des Software Engineering (Bild 3).

Die öffentlich für die Neo4j-Driver bereitgestellten Sprachschnittstellen erleichtern wesentlich die Programmierung. Als Entwickler muss man sich nicht mehr mit sprach- oder schnittstellenspezifischen Details belasten. Vielmehr schuf der Hersteller von Neo4j in Anlehnung an die Vorgehensweise bei anderen SQL-DBMS einen einheitlichen Ansatz für alle grundlegenden Basis-Operationen.

Mit der Kapselung hat Neo Technology ein für die Zukunft stabiles Fundament gelegt, auf das andere Hersteller oder Softwarehäuser mit ihrer eigenen Technologie direkt aufbauen können.

Zudem stehen die Neo4j-Driver in allen spezifischen Software-Verteilungssystemen zur Verfügung: für Java in Maven, Python in PyPI und für JavaScript in NPM; inklusive den zugehörigen Quellen. Ergänzend findet man auf GitHub den zur jeweiligen Programmiersprache gehörigen Quellcode.

Leider verfügt die Sprachschnittstelle von Neo4j über kein eigenständiges Konzept für die satzweise Verarbeitung von Treffermengen. Vielmehr muss man als Programmierer die in der Sprache vorhandenen Konstrukte kennen, auf diese zurückgreifen und sie nutzen. Damit besitzt die Neo4j-Schnittstelle nicht die Mächtigkeit des aus der Welt gängi- ►



Kapselung stellt eine einheitliche Kommunikation über definierte Schnittstellen mit dem Objekt sicher (Bild 3)

ger relationaler DBMS oder Statistik-Pakete bekannten Cursor-Konzepts.

BOLT als Basis für individuelle Produktentwicklung nutzbar

Neo Technology ergänzt den über GitHub verfügbaren Quellcode um ein Beispiel in Python, das aufzeigt, wie man einen Neo4j-Driver auf Basis von BOLT implementiert. In Anlehnung an dieses Python-Beispiel geht man analog in jeder anderen Programmiersprache vor.

Für die Entwicklung eines eigenen BOLT-Treibers benötigt man lediglich ein RAW-Socket, dessen Funktionalität man sich notfalls über eine spezielle Bibliothek besorgt. In der Regel verfügen die heute gängigen Programmiersprachen über ein solches RAW-Socket-API. Als vorteilhaft hat sich die Einheitlichkeit durch Einsatz einer maßgebenden Zuordnung zum nativen Typsystem von Neo4j erwiesen.

Die Implementierung eines BOLT-Treibers ermöglicht quasi den Einsatz eigener Produkte und Programme für Neo4j und macht so die fortschrittliche Graphtechnologie für diese zugänglich. Beispielsweise nutzt die Firma GraphAware die BOLT-Sourcen für das eigene Produkt GraphAware PHPClient.

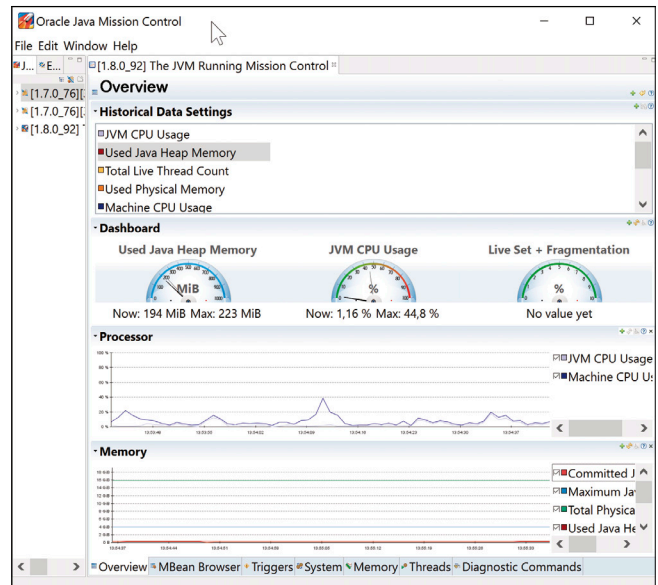
Das Softwarehaus entwickelte als Neo4j-Partner frühzeitig

Listing 1: BOLT-Programmierung

```
// Verbindung zur Graphdatenbank über Treiber
// aufbauen
var driver = neo4j.driver("bolt://localhost:7687",
neo4j.auth.basic("neo4j", "neo4j"));

// Session anlegen
var session = driver.session();

// Session mit Promise-Metapher von JavaScript
// ausführen
session
    .run("MATCH (u:User) RETURN u.name")
    .then(function(result){
// Die Treffermenge der CQL-Abfrage sequenziell
// verarbeiten
        result.records.forEach(function(record) {
            console.log(record._fields);
        });
// Ist der Cypher-Befehl vollständig
// abgearbeitet
// muss die Session geschlossen werden
        session.close();
    })
    .catch(function(error) {
// Fehlerhandlung
        console.log(error);
        session.close();
    });
```



Mit JMC (Java Mission Control) zeichnet man die Ausführung von Java-Programmen auf und kann diese anschließend in einem Analyse-Tool auswerten (Bild 4)

schon während der ersten Meilensteine der Version 3 eine eigene PHP-Sprachschnittstelle als BOLT-Treiber. Ebenso realisierte man auf der Basis des BOLT-Protokolls einen Neo4j-JDBC-Driver.

Die JDBC-Schnittstelle lässt sich aufgrund ihrer breiten Verbreitung auch als Integrations-Plattform für andere Software-Pakete aus dem Umfeld von Office, SQL-Datenbank-Tools, Business Intelligence, Data Warehouse, Reporting oder integrierten Entwicklungsumgebungen (IDE) nutzen. Laut Ankündigung befindet sich außerdem ein Apache-Spark-Connector, der ebenfalls auf dem BOLT-Protokoll basiert, in Entwicklung.

Im Bedarfsfall steht als Profiling-Tool für die Entwicklung eines eigenen BOLT-Treibers Java Flight Recorder (JFR) und eine auf JMX (Java Management Extensions) basierende Console zur Verfügung. JFR sammelt während eines Tests bei der Entwicklung eines BOLT-Treibers Daten zum Zustand der JVM und zeichnet diese auf. Anschließend kann man mit der JMX-Console die resultierenden Aufzeichnungen näher erkunden und zielorientiert auswerten (Bild 4).

Beide Tools (JFR und Auswertungs-Console) gehören als Bestandteil zu Java Mission Control (JMC), das Oracle zusammen mit dem JDK seit Version 1.8 ausliefert. Neo Technology stellt Entwicklern über ein GitHub-Repository eine spezielle Schnittstelle für den Einsatz dieser beiden Werkzeuge zur Verfügung.

BOLT-Sprachtreiber in der Anwendungsentwicklung einsetzen

Der Einsatz eines der vier BOLT-Sprachtreiber setzt dessen Installation voraus; die man am einfachsten über einen der typischen Dependency Manager für C#.NET (NuGet), Java (Maven), JavaScript (NPM) oder Python (pip) durchführt. Seitens der Anwendungsentwicklung findet die Kommunikati-

on mittels BOLT in allen Sprachen gemäß einem einheitlichen Muster statt: Zu Beginn baut man eine Verbindung zur Graphdatenbank auf, identifiziert sich mit Benutzerkennung und Kennwort und erzeugt so ein Neo4j-Driver-Objekt.

Anschließend legt man ein neues Session-Objekt für die Ausführung von Cypher-Befehlen an. Das Session-Objekt liefert ein Objekt zurück, das die Ergebnismenge eines Cypher-Befehls repräsentiert.

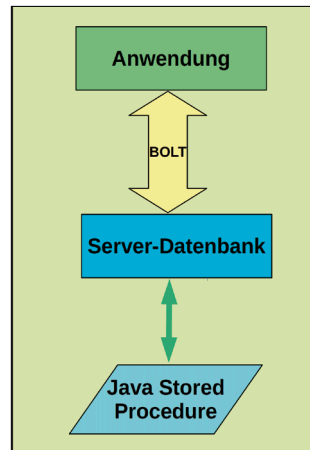
Gegebenenfalls führt man die Cypher-Befehle einer Session innerhalb einer Transaktion aus. Bei einer Datenbank-Transaktion müssen im Erfolgsfall die Daten persistent gemacht und im Fehlerfall die Transaktion zurückgesetzt werden.

Nach der Verarbeitung der Ergebnismenge muss die angelegte Session wieder geschlossen werden. Jede Session beansprucht interne Ressourcen des Neo4j-Treibers; insofern muss diese auch im Fehlerfall wieder geschlossen werden, um die beanspruchten Ressourcen wieder freizugeben.

Reference-Manual für Programmierer

Für alle offiziell unterstützten Sprachen stellt Neo Technology jeweils ein neues Reference-Manual für Programmierer bereit; zudem findet man auf GitHub die zugehörigen Treiber-Realisierungen inklusive Demo-Quellcode. Das Beispiel in [Listing 1](#) basiert auf dem Einsatz des Neo4j-Drivers in JavaScript.

Die über GitHub von Neo Technology verfügbare Handlungsanleitung erläutert ganz detailliert die Arbeitsweise von BOLT. Sie beschreibt auch, wie man im Fehlerfall Debugging durchführt und eigene Tests aufsetzt. Sollten im Netzwerk beim Einsatz von BOLT wider Erwarten doch Engpässe vorkommen, was die Performance eigener Anwendungen betrifft, so können diese Erläuterungen für Rückschlüsse hilf-



Eine Client-Anwendung greift über das BOLT-Protokoll auf die Server-Datenbank zu, um auf dem Server eine Stored Procedure auszuführen (Bild 5)

reich sein. Somit lassen sich die Ursachen eventueller Probleme seitens der Performance einer Anwendung lokalisieren und darauf aufbauend die erforderlichen Lösungswege erkunden. Dazu müssen sicherlich Netzwerk-Analysiker und Programmierer eng zusammenarbeiten.

Built-in und gespeicherte Prozeduren in Cypher neu verfügbar

Bisher kannte Cypher eigenständige Funktionen wie Prädikate, Skalare, Collections oder mathematische Funktionen, die wie in einer Programmiersprache üblich durch einen Aufruf verschiedene Rückgaben liefern.

Neo4j Version 3 führt jetzt erstmals wirkliche Prozeduren ein, sogenannte Built-in und Stored Procedures, die im Unterschied zu einer Funktion nicht zwingend eine Rückgabe liefern. Während die Ersteren der Datenbankhersteller fertig programmiert, werden

die Stored oder gespeicherten Procedures extern selbst entwickelt. Die Ausführung beider Prozedurtypen erfolgt über ein neues `CALL`-Statement in Cypher.

Damit beschreitet Neo Technology einen konsequenten Weg für deren direkte Integration in das DBMS; bei Wahl eines spezifischen Konstrukts für die jeweilige Sprache hätte man dieses Ziel nicht erreicht.

Built-in Procedures umfassen zum einen Prozeduren für die Abfrage des Data Dictionarys einer Graphdatenbank, um so Informationen beispielsweise über die Namen von Labels, Beziehungstypen, Property Keys, Indizes oder Constraints zu erhalten.

So gibt zum Beispiel `CALL dbms.procedures` alle im DBMS bekannten Prozeduren aus. Entsprechend erzeugt ein Aufruf von `ListComponents` mittels `CALL dbms.components` eine Liste aller DBMS-Komponenten und ihrer Versionen. Abfragen von Informationen des JMX nach Domain und Name erfolgen mittels der Prozedur `QueryJmx` durch `CALL dbms.queryJmx(query)`. `AlterUserPassword` ändert das Benutzerkennwort durch `CALL dbms.changePassword(query)`.

Stored Procedures, manchmal auch User-defined Procedures genannt, ergänzen die bereits verfügbaren Neo4j-Server-Extensions, beides realisiert man als Plug-in; jedoch lässt sich nur die Funktionalität der Ersteren in Cypher aufrufen.

Derzeit unterstützt die neue Version 3 ausschließlich Java Stored Procedures; allerdings kann man seitens der Entwicklung deren Funktionalität in jeder auf der JVM verfügbaren Sprache (JavaScript, Clojure, Groovy, Scala, JRuby, Jython) programmieren. Wie bei anderen DBMS auch erfolgt der Aufruf einer gespeicherten Prozedur innerhalb einer Anwendung. Für die Kommunikation mit der Server-Datenbank nutzt die Anwendung beim Zugriff auf die Stored Procedure automatisch das neue BOLT-Protokoll ([Bild 5](#)), insbesondere reduziert BOLT die Kommunikationslast bei der Ausführung der Stored Procedure.

Der Realisierungsaufwand einer Java Stored Procedure ►

Cypher-Kompatibilität in Neo4j-Version 3.0.x

Das Cypher-System 3.0 im aktuellen Neo4j-DBMS unterstützt die Cypher-Versionen 1.9, 2.2 und 2.3.

Danach sind alle CQL-Befehle aus der Sprachversion 2.x ohne Änderungen in der Neo4j-Version 3.0.x lauffähig.

Um aus Gründen der Kompatibilität für die Cypher-Befehle eine bestimmte Version auszuwählen, geht man wie bereits früher in den Vorgängerversionen von Neo4j vor. So lässt sich in der Datei `neo4j.conf` über den Konfigurationsparameter `cypher.default_language_version` eine bestimmte Version durch Auswahl festlegen: 2.3, 3.0, `default`. Weiterhin legt man über die erste Zeile einer Cypher-Befehlsfolge anhand des Statements `CYPHER version_number` eine gewünschte Version fest, zum Beispiel die Version 1.9 mittels der Befehlsklausel `CYPHER 1.9`.

mit Gradle oder Maven hält sich in Grenzen, zumal Neo Technology Templates und Beispiele über spezielle GitHub-Repositories bereitstellt. Dabei erfolgen Builds über das Maven-System; für das Packaging sollte man das Maven-Shade-Plug-in einsetzen, da dieses alle Abhängigkeiten der Klassen/Ressourcen des Projekts für das JAR berücksichtigt.

Im Java-Quellcode kennzeichnet man alle Stored Procedures mit einer `@Procedure`-Annotation; bei allen Prozeduren schreibenden Zugriffs auf die Graphdatenbank vermerkt man zusätzlich eine `@PerformsWrites`-Annotation. Nutzt man das Java-Testing-Framework JUnit zusammen mit dem seit Neo4j Version 2.1.6 verfügbaren Testing Harness, vereinfacht dies wesentlich alle Integrationstests.

APOC-Sammlung bereits für Version 3 verfügbar und zusätzlich erweitert

Anfangs stand das Akronym APOC für A Package Of Components, inzwischen verwendet man die Abkürzung aber auch für Awesome Procedures On Cypher: APOC stellt eine Sammlung von Prozeduren dar, die man standalone über `CALL apoc.procedure()` aufruft.

APOC kann aber auch als eine Erweiterung von Cypher angesehen werden. In diesem Fall führt APOC via Kommandozeile dynamische Cypher-Befehle aus, sodass sich APOC direkt in Cypher integriert. Das aktuelle Release umfasst inzwischen knapp 200 Prozeduren und steht zeitgleich mit der neuen Neo4j-Version über GitHub als JAR-Datei oder im Quellcode zur Verfügung.

Die Sammlung an APOC-Prozeduren wurde nicht nur an die Neo4j-Version 3 angepasst, sondern zusätzlich um weitere Features erweitert: Bit-Operationen, verschiedene Graph-Algorithmen, Erzeugen des Meta-Graphen, Laden von Daten in die Graphdatenbank mittels JDBC, JSON, XML oder CSV, räumliche Funktionen (Spatial Operators) sowie Freitextsuche.

So enthält APOC eine Implementierung des PageRank-Algorithmus der Google-Suche, um die Bedeutung der Knoten eines Graphen zu messen. Für Änderungen im Datenmodell des Graphen kann man Prozeduren zum Refactoring nutzen. Auch steht eine Schnittstelle zur Suchmaschine Elasticsearch zur Verfügung.

Näheres zur Installation und dem Einsatz von APOC erläutert ein über das Web zugängliches Benutzerhandbuch. Die Entwicklung von APOC findet über ein eigenes, von Neo4j unabhängiges Projekt als WIP (Work in Progress) statt. Besonders nützlich beim Arbeiten mit APOC erweist sich das direkt in APOC integrierte Hilfesystem. So gibt `call apoc.help("apoc")` eine Liste der Prozedurnamen mit einer kurzen Funktionsbeschreibung aus. Um

Produkte für den Aufbau einer Docker-Infrastruktur

Im Umfeld von Docker entstand eine Vielzahl von Produkten und Services:

- Docker Hub: ein Cloud-Service mit Repository-Features für öffentliche und private Inhalte, um die Entwicklung in Teams zu unterstützen.
- Docker Datacenter (DDC): eine Plattform für Management und Deployment von Containern im Intranet.
- Docker Cloud: ein Cloud-Service für Management und Deployment von Containern über das Internet.
- Docker Toolbox: eine Sammlung von Spezialwerkzeugen (Docker Machine, Docker Swarm, Docker Compose, Docker Registry, Docker Engine, Kitematic) für Mac OS X und Windows zur Unterstützung von Entwicklung, Bereitstellung und Betrieb moderner, verteilter Anwendungen.

die Arbeitsweise einer APOC-Prozedur mit dem Hilfesystem gezielt zu erkunden, ruft man `call apoc.help("prozedur-name")` auf. Für die Automatisierung in Rechenzentren eignet sich das speziell in APOC realisierte Job-Management.

Neo4j Browser Sync macht den Arbeitsplatz über das Web verfügbar

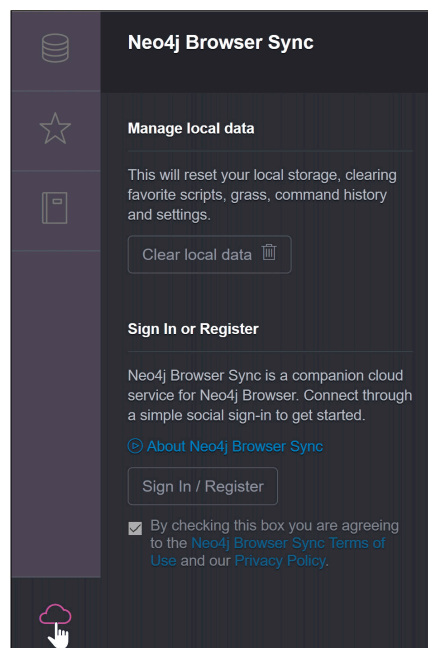
Frühzeitig erkannte Neo Technology den Wunsch vieler Entwickler und Endbenutzer nach Zugriffsformen auf Graphdatenbanken über das Web und führte in Version 2 das Neo4j Browser-Tool ein. Mit der Version 3 stellt der Hersteller nun ein neues Feature zur Synchronisierung von individuellen

Einstellungen dieser interaktiven Arbeitsumgebung bereit: Neo4j Browser Sync.

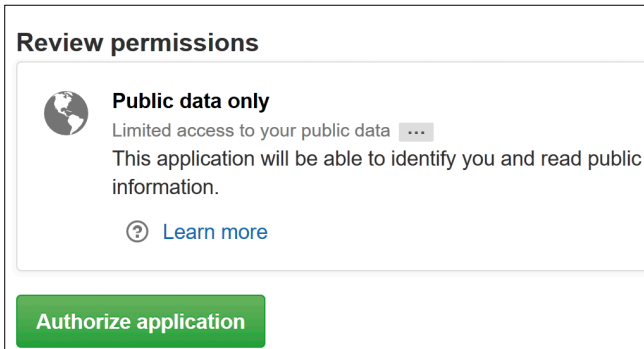
Dieser zusätzliche Service des Neo4j Browser-Tools synchronisiert über die Cloud selbst erstellte Queries, individuelle Skripts und Stylesheets für Anzeige und Bearbeitung von Graphen. Damit stehen Favoriten und das Styling samt Einstellungen der Arbeitsplätze sogar über unterschiedliche Webbrowser hinweg zur Verfügung.

Neo4j Browser Sync setzt für seinen Benutzer einen der folgenden Accounts voraus: GitHub, Google oder Twitter; zusätzlich muss man bei der erstmaligen Anmeldung den Nutzungsbedingungen zustimmen.

Diesen Service, kurz Browser Sync genannt, erreicht man am schnellsten durch Klick auf das kleine magentafarbene Wolken-Symbol (Bild 6) links unten in der Tool-Leiste des Neo4j Browsers oder durch Eingabe des Kommandos `:play neo4j sync`. Nach Klick auf das



In der linken Tool-Leiste des Neo4j Browsers meldet man sich für den neuen Service Browser Sync an (Bild 6)



Zur Identifikation des Benutzers benötigt der neue Service Browser Sync eine Zugriffsberechtigung auf vorhandene Benutzerdaten (Bild 7)

Auswahlkästchen zur Bestätigung der Nutzungsbedingungen öffnet sich ein Sign-in-Dialogfenster für die eigene Anmeldung über einen bestehenden GitHub-, Google- oder Twitter-Account. Alternativ legt man über dieses Dialogfenster nach Auswahl einen Neuzugang zu einem der drei angebotenen Account-Typen an und führt anschließend ein Sign-in aus.

Bei der erstmaligen Anmeldung über einen der drei Account-Typen muss der Neo4j-Dienst Browser Sync für den Zugriff auf die Daten des jeweiligen Accounts autorisiert werden. Browser Sync benötigt lediglich eine eingeschränkte Berechtigung für Zugriffe auf die öffentlichen Account-Daten des Nutzers (Bild 7).

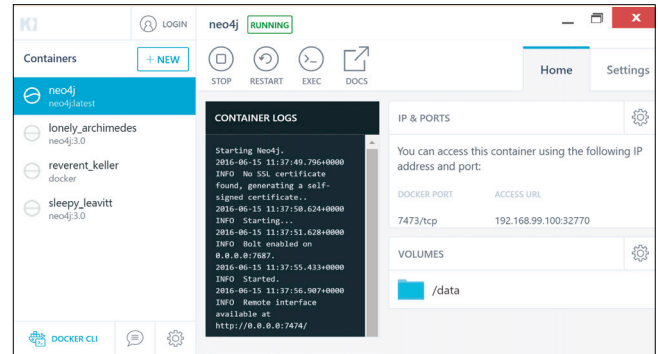
Nach stattgefundenem Sign-in erscheint in der linken Tool-Leiste des Neo4j Browsers anstatt des kleinen magentafarbenen Wolken-Symbols ein grünes Cloud-Services-Symbol. Dabei erfolgt eine Synchronisierung der drei Bereiche *Favorites*, *Folders* und *GRASS*. Wählt man über die darunterstehenden Auswahllisten eine Version aus, erscheint eine Restore-Schaltfläche. Ein Klick auf *Restore* setzt den zugehörigen Bereich auf die ausgewählte Version zurück.

Der Browser-Sync-Service in seiner aktuellen Realisierung erzeugt aus jeder vorgenommenen Änderung in einem der drei Bereiche (*Favorites*, *Folders*, *GRASS*) automatisch eine neue Version. Diese zeigt der Bereich *Cloud Services* in der betroffenen Auswahlleiste auch an.

Die Bezeichnung einer Version besteht aus dem Datum, einer Uhrzeit mit Zeitzone sowie dem Namen des Webbrowsers, von dem sie erzeugt wurde. Die Anzahl der maximal verwalteten Versionen beschränkt das Browser-Sync-Feature derzeit auf fünf. Wünschenswert wäre es, wenn der Benutzer im Neo4j Browser die Anlage einer Version selbst anstoßen und für sie einen eigenen, selbst gewählten Namen vergeben könnte. Die derzeitigen Bezeichnungen der Versionsnamen sind leider nicht selbsterklärend; ein Benutzer würde sicherlich sprechende Versionsnamen wählen.

Container-Engine Docker nicht nur für Packaging und Deployment

Mit Version 2.3 von Neo4j stellte Neo Technology ein erstes Beta-Release für ein Docker-Image der Öffentlichkeit vor.



Mit Kitematic erledigt man alle Aufgaben rund um Docker, auch die Installation und Verwaltung von Containern mittels einer Schnittstelle zum Docker Hub (Bild 8)

Dieses wurde anschließend als offizielles Produkt zusammen mit der Firma Docker auch für die neue Version 3 weiterentwickelt und freigegeben. Unter einem Image versteht das Docker-System eine portable Abbildung, auch Schnappschuss genannt, eines Containers, der als konfigurierbare, abgeschlossene Einheit eine Anwendung mit allen notwendigen Bestandteilen enthält. Docker-Container entsprechen gewissermaßen einer Infrastruktur- und Ausführungsebene für durchgängige, plattformunabhängige Verteilung und Betrieb von Software.

Aufbau und Zusammensetzung eines Containers, also dessen Bestandteile, beschreibt in Form eines Bauplans ein sogenanntes Dockerfile. Docker-Images kommunizieren normalerweise über Network-Ports, zusätzlich ist mittels Linking eine geschützte Kommunikation möglich. Generell zielt das Neo4j-Docker-Image auf ein leichteres und schnelleres ►

Docker Toolbox – das Fundament im Docker-Arsenal

Container kapseln Funktionen/Services in eigenständige Einheiten für die hardwareunabhängige Ausführung.

Im Fokus der Docker-Welt von Mac OS X und Windows steht die Docker Toolbox, eine Sammlung von Werkzeugen für die Docker-Technologie:

- Docker Machine: unterstützt Verwaltung, Einrichtung und Betrieb eines Docker-Systems und einer Docker Engine.
- Docker Compose: für Aufbau und Management verteilter Anwendungen, die man aus einzelnen Docker-Containern zu einer Einheit zusammensetzt.
- Docker Engine: das Herz der Docker-Infrastruktur, erzeugt Docker-Container und führt sie aus.
- Docker Swarm: baut über Clustering eine Gruppe von Docker Engines zu einer Einheit als virtuelle Docker Engine zusammen.
- Docker Registry: übernimmt die Datenhaltung und Verteilung der Docker-Images.
- Kitematic: eine benutzerfreundliche Desktop-Anwendung für die komplette Docker-Infrastruktur mit Verbindung zu Docker Hub.

Deployment des DBMS oder eines kompletten Datenbanksystems auf allen für Docker verfügbaren Systemplattformen ab. Sobald ein Image sich als Datei auf einem Docker-System befindet, lässt es sich mittels Virtualisierung als eigenständige Instanz (isoliert und damit sicher von allen anderen) starten und ausführen. Die Installation eines Docker-Systems mit seinen Komponenten erfolgt abhängig vom Betriebssystem.

Aktuell empfiehlt es sich, unter Windows die sogenannte Docker Toolbox zu nutzen, da sie die veraltete virtuelle Maschine Boot2Docker direkt ablöst. Zusätzlich enthalten sind neben einem Docker-System der GUI-Client Kitematic (derzeit für Windows noch im Alpha-Release) (Bild 8) sowie der Startbefehl Docker Quickstart Terminal, den die Installations-



Kitematic bietet dem Benutzer verschiedene Releasestände eines Docker-Images zur Auswahl an (Bild 9)

routine der Docker Toolbox als Shortcut auf dem Desktop ablegt. Als virtuelle Maschine kommt jetzt VirtualBox von Oracle zum Einsatz; diese richtet die Docker Toolbox zusammen mit der GNU bash-Shell automatisch ein.

Die Installation macht zudem eine interaktive Docker-Shell (docker-machine) sowie eine Docker-Engine (docker) verfügbar, um Images anzulegen, zu verwalten, aus Docker Hub zu beziehen und auszuführen. Beide, Docker-Shell und Docker-Engine, enthalten (analog bash) ein integriertes Hilfesystem, das Beschreibungen über Befehle, deren Syntax und Ausführung am Terminalfenster anzeigt:

docker-Shell und Docker-Engine, enthalten (analog bash) ein integriertes Hilfesystem, das Beschreibungen über Befehle, deren Syntax und Ausführung am Terminalfenster anzeigt:

```
# Hilfeinformation zum Kommando run abfragen
# und anzeigen
docker run --help
```

```
# Hilfeinformation zum Kommando start abfragen und
# anzeigen
docker-machine start --help
```

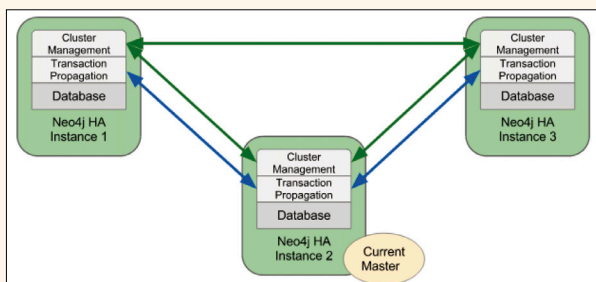
Skalierbarkeit und Performance – Neo4j HA–Package

Die Enterprise Edition von Neo4j enthält ein High-Availability-Package (HA), um hohe Verfügbarkeit und Performance zu gewährleisten.

Neo4j HA realisiert eine skalierbare, fehlertolerante Datenbank-Architektur mittels Clustering. Im Fehlerfall bietet die Datenbank ihren Endbenutzern stets die volle Funktionalität, was das Lesen und Schreiben auf die Datenbank betrifft.

Neo4j HA setzt sich aus einem Single-Master und mehreren Slaves zusammen: Die Slaves basieren auf Replikationen der Master-Datenbank (Master-Slave-Replikation). Beide Knoten, Master und Slave, akzeptieren sowohl Lese- als auch Schreib-Anforderungen. Ein Slave arbeitet Schreib-Operationen durch Synchronisierung mit dem Master ab, um Konsistenz sicherzustellen. Updates eines Slave werden asynchron abgearbeitet, sodass diese nicht sofort für alle anderen Slaves sichtbar sind.

Neo4j HA stellt eine horizontal skalierbare Read-mostly-Architektur dar; das DBMS bietet im HA-Betrieb viel höhere Durchsätze beim Lesen der Daten, die eine Single-Instanz alleine nie erreichen würde.



Synchronisation eines Slaves findet zuerst mit dem Master statt; dieser aktualisiert anschließend alle anderen per Push. Im Fall eines Konflikts entscheidet der Master über dessen Lösung


Neo Technology hat auf Docker Hub zwei verschiedene Repository-Systeme eingerichtet, um das Datenbanksystem sowohl für Produktionssysteme als auch in der Entwicklung einsetzen zu können. Für jede der beiden Umgebungen (Betrieb und Test) benötigt man einen eigenständigen Docker-Container: ein offizielles Repository von Neo4j für Produktions- und Entwicklungszwecke, sowie ein zweites, experimentelles Repository, dessen jeweilige Releasestände von Neo4j sich für Entwicklung und Tests eignen. Zudem findet man in diesen zwei Repositories für beide ausgelieferten Editionen Community und Enterprise die zugehörigen Docker-Files.

Die Installation eines Neo4j-Containers in Docker erledigt man entweder über die Docker-Engine docker oder den GUI-Client Kitematic. Mittels der Docker-Engine führt man einen `run`-Befehl aus; dieser prüft, ob das gewünschte Image bereits vorhanden ist. Sollte dies noch nicht der Fall sein, erzeugt die Docker-Engine anhand weiterer Parameter eine virtuelle Maschine, sucht das zugehörige Docker-Image im Docker-Hub-Repository, lädt es herunter, konfiguriert es und passt es seiner Systemumgebung für die anschließende Ausführung anhand der übergebenen Optionen an:

```
# Neo4j als Image in der aktuell laufenden virtuellen
# Maschine installieren und starten
# die beiden Ports für Neo4j Browser (7474)
# und für das BOLT-Protokoll (7687) bekannt geben
docker run \
  --publish=7474:7474 --publish=7687:7687 \
  --volume=$HOME/neo4j/data:/data \
  neo4j:3.0
```

Nach dem Start von Kitematic versucht das Programm, eine Verbindung zum laufenden Docker-System aufzubauen.

IP & PORTS



You can access this container using the following IP address and port:

DOCKER PORT	ACCESS URL
7473/tcp	192.168.99.100:32773
7474/tcp	192.168.99.100:32772
7687/tcp	192.168.99.100:32771

Der GUI-Client von Docker zeigt die eingerichteten physischen Adressen der Docker-Ports auf dem Hostsystem an (Bild 10)

Sollte noch kein Docker-System gestartet sein, so erfolgt eine Informationsmeldung, und ein Klick auf *Use VirtualBox* initialisiert eine Linux VM und startet Docker.

Kitematic führt in der linken Liste alle bereits in der Docker VM installierten Container auf. Unterhalb der Fensterleiste öffnet sich über die *New*-Schaltfläche ein Eingabefeld zum Durchsuchen des Docker-Hub-Repositorys.

Gibt man dort als Suchkriterium *Neo4j* ein, zeigt Kitematic alle Treffer mit einer Beschreibung an. Ein Klick auf *CREATE* lädt das ausgewählte Docker-Image herunter und installiert es in einen Container. Über das Drei-Punkte-Icon der Docker-

NEO4J_VERSION	3.0.2	ⓧ
NEO4J_EDITION	enterprise	ⓧ
NEO4J_DOWNLOAD_	7e3cb1cb29c51ac267e57ec3d12c425b...	ⓧ
NEO4J_DOWNLOAD_	http://dist.neo4j.org	ⓧ
NEO4J_TARBALL	neo4j-enterprise-3.0.2-unix.tar.gz	ⓧ
NEO4J_URI	http://dist.neo4j.org/neo4j-enterprise-	ⓧ
		⊕

Mit einem Klick auf das grüne Plus-Symbol definiert man neue Übergabeparameter für den laufenden Neo4j-Container (Bild 11)

Kachel erreicht man ein Dialogfenster zur Auswahl aus verschiedenen Releaseständen (Bild 9).

Neo4j über Docker individuell für Betrieb und Entwicklung konfigurieren

Nach Installation führt die Docker-Engine mittels des *run*-Befehls das Image aus; in unserem Beispiel startet die virtuelle Maschine von Docker die Neo4j-Datenbank.

Um auf eine Graphdatenbank über einen Neo4j Browser zugreifen zu können, muss man die IP-Adresse mit den zugehörigen Ports der Docker-Maschine kennen. Leider erscheint derzeit am Bildschirm die falsche IP-Adresse für den Zugriff auf die Graphdatenbank über das Neo4j Browser-Tool. Den korrekten URL erhält man über den Befehl `docker-machine inspect default`; er gibt für das Docker-System mit dem Namen *default* im Abschnitt *IPAddress* die individuell zugeordnete IP-Adresse (192.168.99.100) aus. Alternativ findet man in Kitematic über den Registerreiter *Home* im Bereich *IP & PORTS* oder über das Register *Settings* im Bereich *Ports* den Docker-Port 7474 für den korrekten Zugriffs-URL (Bild 10).

Statt durch Eingabe von `http://localhost:7474` im Webbrowser erreicht man die Graphdatenbank im Neo4j Browser beispielsweise jetzt über `http://192.168.99.100:32769`. Für die Kommunikation der Anwendungen mit der Datenbank benötigt man bei Einsatz des BOLT-Protokolls kein Linking über Verzeichnisse mit dem Hostsystem.

Neo4j hat beim Start für das BOLT-Protokoll einen entsprechenden Docker-Port 7687 eingerichtet. Den zugehörigen Zugriffs-URL findet man über den GUI-Client Kitematic in den beiden zuvor genannten Ausgabebereichen.

Für alle im Rahmen einer Neo4j-Installation anfallenden Aufgaben bietet Docker eine entsprechende Unterstützung, die Neo Technology über spezielle Befehle für die Docker-Engine implementiert hat. Diese direkte Docker-Unterstützung unterteilt sich in die nachfolgenden Aufgabenbereiche:

- Neo4j-Konfiguration: Ändern der vorgegebenen Standardwerte der Datenbankkonfiguration wie Umgebungsvariable, Authentifizierung, Kennwort, Cache-, Heap-Size und weitere.
- Neo4j HA: Einrichtung der benötigten Kommunikationskanäle für den Betrieb der Datenbank als High-Availability- ►

Update/Patch – Upgrade – Migration, Portierung

Als Kennzeichnung für den Aufwand einer Umstellung von Software verwendet man in der Informatik die folgenden zentralen Begriffe:

- **Update:** Ein Update zielt auf die Aktualisierung eines bereits eingesetzten Systems ab, mit der Zielsetzung, kleinere Verbesserungen oder Fehlerkorrekturen zu nutzen.
- **Patch, Hotfix, Bugfix, Service-Release, Service-Pack:** Diese Begriffe rücken die Behebung von Fehlern eines eingesetzten Produkts in den Mittelpunkt. Die Fehlerbeseitigung sollte in der Regel automatisch erfolgen.
- **Upgrade:** Dabei handelt es sich um die vollständige Ersetzung einer Systemversion durch eine andere, neuere Version desselben Produkts. Im Unterschied dazu ersetzt ein Downgrade eine aktuelle Version durch eine ältere, in der Regel zuvor bereits eingesetzte Version. Dieser Vorgang benötigt mehr Aufwand als ein einfaches Update oder ein Patch.
- **Migration/Portierung:** Bezeichnet die Umstellung einer kompletten Systemlandschaft (Systemsoftware und Anwendungen). Von Portierung spricht man vor allem beim Umstieg auf ein anderes, bisher nicht eingesetztes Produkt. Aufgrund des Produktwechsels rechnet man hier mit dem höchsten Aufwand an Ressourcen.

Database

Version: 2.3.2
 Location: C:\Program Files\neo4j-community-2.3.2\data\graph.db
 Size: 468.61 KiB
 Information  sysinfo

Über die Database-Information des Neo4j Browsers lässt sich schnell die Version des aktuell installierten DBMS sowie der Ablageort und Dateiname der Graphdatenbank ermitteln (Bild 12)

ty-Cluster inklusive Online-Backup und Monitoring (nur in der Enterprise Edition von Neo4j verfügbar).

- User-Defined Procedures: Installation der JARs für User-Defined Procedures über ein *plugins*-Volume.
- Neo4j Shell: Starten einer Neo4j Shell innerhalb eines Containers über die Docker-Engine.
- TLS: HTTPS-Kommunikation durch Einrichten eines *ssl*-Volumes mit *key*- und *cert*-Datei.

Zusätzlich erreicht man alle Einstellungen des Neo4j-Containers in Kitematik über den Registerreiter *Settings* mit den Bereichen *General*, *Ports*, *Volumes* und *Advanced*. Dort lassen sich die standardmäßig vorgegebenen Werte ändern oder neue Einstellungen für Variablen, Port-Adressen, Volumes oder Ähnliches definieren (Bild 11).

Achtung: Um Vorgaben für einen Container zwischen den Docker-Sessions zu sichern, muss man die SAVE-Schaltfläche betätigen. Die so vorgegebenen Einstellungen übernimmt

Gespeicherte Prozeduren besitzen Vor- und Nachteile

Programmierer legen gespeicherte Prozeduren im Data-Dictionary der jeweiligen Datenbank ab – nicht im DBMS selbst.

Eine gespeicherte Prozedur umfasst komplette Abläufe von Anweisungen an das Datenbanksystem, die zwar der Client in der Anwendung startet, aber der Server ausführt.

Mit gespeicherten Prozeduren sind wichtige Vorteile wie Mehrfachverwendbarkeit, Plattformunabhängigkeit, zentrale Wartbarkeit, Vermeidung von unnötigem Netzwerkverkehr zwischen Client und Server, erhöhte Sicherheit und Ähnliches verbunden. Leider mit einem wesentlichen, nicht zu unterschätzenden Nachteil: Bei einer Vielzahl paralleler Zugriffe auf die Datenbank kann es auf dem Server zu einem Kapazitätsengpass kommen. Möchte man eine gespeicherte Prozedur in einer stark frequentierten Umgebung einsetzen, so sind ausführliche Lasttests nicht nur mit DBMS und Datenbanksystem, sondern in der kompletten Systemumgebung unter Realbedingungen durchzuführen.

config-migrator von 2.x.a auf Version 3.0.x

Neo Technologie hat den config-migrator als Werkzeug entwickelt, um die Einstellungen der Konfiguration einer 2.x.a-Datenbank automatisch in eine neue Neo4j-Umgebung für die Version 3.0.x zu übertragen.

Als Parameter übergibt man dem Java-Programm config-migrator den Ablageort der Datenbank aus Version 2.x.a und das Zielverzeichnis mit der neuen Version 3.0.x-Datenbank. Bei der Ausführung sind eventuelle Warnhinweise zu beachten. Zudem sollten automatisch durchgeführte Einstellungen der Konfiguration anhand der ermittelten Vorgaben überprüft werden.

die Docker-Engine zum Zeitpunkt der Ausführung des jeweiligen Containers. Fehlende Einstellungen eines ausgewählten Containers zeigt der GUI-Client Kitematik im *Home*-Register unmittelbar neben der Container-Liste in seiner Log-Ausgabe an.

Schrittweise Umstellung für den On-Premise-Einsatz der neuen Version 3

Eine Migration auf die neue Version 3 von Neo4j, seitens Neo Technology treffenderweise Upgrading genannt, erfordert im ersten Schritt die Schaffung der notwendigen Voraussetzungen. Im Fall von Neo4j benötigt man für das Upgrade auf die aktuellste Version lediglich den Einsatz einer bestimmten älteren Vorgängerversion.

Der englische Fachbegriff Upgrading weist direkt auf die Notwendigkeit eines Umstellungsprozesses hin. Er umfasst verschiedene Arbeitsschritte, die der Hersteller bereits vorkonstruiert und deren Reihenfolge man als Anwender kon-

Leistungsfähige Cursor-Programmierung

Unter einem Cursor versteht man eine Datenstruktur, die analog einem aus Textverarbeitungen bekannten Cursor durch das Dokument wandert, um so eine Menge von Daten zu verarbeiten.

Ein Datenbank-Cursor navigiert durch die einzelnen Sätze einer Treffermenge; normalerweise erfolgt dies über den Cursor sequenziell hintereinander (Forward-only Cursor).

Relationale DBMS mit SQL-Schnittstellen oder einige Statistik-Bibliotheken verfügen auch über eine Funktion, um die Treffermenge mit dem Cursor rückwärtsgerichtet vom Ende zum Anfang zu durchlaufen (Static oder Scrollable Cursor).

Server-Side Cursors arbeiten im Unterschied zu Client-Side Cursors auf der Serverseite, sodass für ein Anzeigen der Daten auf dem Client nur die betroffenen Sätze zu übertragen sind, und mit einem Hold-Cursor kann man die Treffermenge einer Abfrage auch nach deren vollständigen Verarbeitung über den Cursor noch zugänglich und für das Programm verfügbar halten.

Docker für moderne verteilte Anwendungen

Container eignen sich besonders für die Umsetzung von Microservices, die kleine, weitgehend entkoppelte Aufgaben erledigen und sich sprachunabhängig realisieren lassen.

Microservices als Architekturmuster zielen darauf ab, komplexe Anwendungen aus kleinen, unabhängigen Prozessen zusammenzusetzen, die unabhängig von der Realisierungssprache untereinander kommunizieren.

Anwendungen auf Basis von Microservices sollen im Sinne einer dienstorientierten Architektur (SOA) leichter änderbar, erweiterbar und skalierbar sein als monolithische Systeme, selbst bei stetig wachsenden und sich permanent ändernden Anforderungen.

sequent einhalten muss. Wie bei allen vorherigen Upgrades auf eine neue Neo4j-Version führt man die Umstellung auf Version 3 des DBMS in mehreren Einzelschritten durch.

Version 3 von Neo4j setzt erstmals die Installation der Java-8-Technologie (OpenJDK, Oracle, IBM) voraus. Die ausführbaren Desktop-Installer der Datenbank enthalten bereits eine passende Java 8 JRE.

Im ersten Schritt prüft man, welche Versionen des DBMS von Neo4j auf dem jeweiligen System aktuell eingesetzt werden und wo sich die zugeordneten Graphdatenbanken befinden. Dieser Prüfprozess stellt im Fall eines Systems mit verschiedenen Releaseständen des DBMS sicher, dass man tatsächlich mit der korrekten Version arbeitet.

Um eine aktuell installierte Neo4j-Version und die ihr zugeordneten Graphdatenbanken zu ermitteln, gibt es verschiedene alternative Vorgehensweisen. Am schnellsten er-

ledigt man dies nach Start des Neo4j-Servers abhängig vom Betriebssystem wie folgt:

- Windows: Man öffnet eine Windows PowerShell, importiert die Neo4j-Module und erhält über den Modul-Befehl *Get-Neo4jServer* die aktuell installierte Neo4j-Version angezeigt.
- Unix/Linux/Mac OS X: Über ein Terminal-Fenster öffnet man eine Neo4j-Shell und führt den Shell-Befehl *dbinfo -g Kernel* aus. Dieser liefert als Ausgabe im Feld *KernelVersion* die Version des installierten DBMS sowie den Ablageort der Graphdatenbank im Feld *StoreDirectory*. Anhand dieser Abfrage der Kernel-Management-Bean erfährt man zusätzlich noch den Dateinamen der Graphdatenbank.
- Alle Betriebssysteme: Meldet man sich über den Neo4j Browser am DBMS an, so erscheint direkt unterhalb als Ausgabe von *:play start* die von ihm verwendete Datenbank-Version. Klickt man alternativ auf das Symbol mit den drei Kreisen ganz links oben, so erscheint rechts davon der Ausgabebereich *Database Information*. Dort erfährt man im unteren Bereich *Database* den Ablageort der Graphdatenbank sowie ihren Dateinamen (Bild 12).

Im nächsten Schritt stellt man fest, welche Upgrade-Pfade der Hersteller für die aktuell gewünschte Neo4j-Version unterstützt. Diese Information findet man in der System-Dokumentation der neuen Neo4j-Version, auf die migriert werden soll.

Automatisches Upgrade der Daten

Im Fall der neuen Version 3 führt Neo Technology nur für den letzten aktuellen Releasestand einer bestimmten Version 2.x automatisch ein Upgrading der Daten durch. Will man beispielsweise von Version 2.x auf die 3.0.y umstellen, so muss man vorher auf das letzte Update (den letzten Releasestand) dieser 2.x-Version ein Upgrade vornehmen. ►

Index-free Adjacency mit wenigen Erweiterungen

Als Datenbank unterstützt Neo4j gerichtete Graphen (Digraphen) mit Properties und Labels, die ergänzend Index-free Adjacency bieten.

Das zugehörige Metamodell bietet keine weiteren Erweiterungen der Graphentheorie wie Multi- oder Hypergraphen und kennt auch keine Gewichte oder Färbungen. Durch Index-free Adjacency enthält jedes Element der Graphdatenbank eine direkte Verbindung zu seinem benachbarten Element. Zugriffe des DBMS auf das Nachbarelement erfolgen extrem schnell (nativ), ohne zusätzlich auf einen Index zurückgreifen zu müssen. Eventuell notwendige Erweiterungen der Graphentheorie bildet man in Neo4j über spezielle Vorgehensweisen und damit verbundene Konventionen ab.

Hypergraphen kommen vor allem in Netzwerken, CAM-Systemen, der kombinatorischen Optimierung, bei Spieltheorie, Data Mining, maschinellem Lernen, Molekularbiologie und anderen Anwendungsfällen zum Einsatz.

Portabilität durch sparsame Virtualisierung

Die Open-Source-Software Docker stellt eine Bereitstellungs- und Ausführungsumgebung für Docker-Images über Linux Container (LXC) dar.

Ein Container isoliert eine Ausführungsumgebung mittels leichtgewichtiger Virtualisierung. Über eine Image-Datei lässt sich ein Container einfach transportieren und installieren. Zudem gewährleisten Container eine Trennung der von ihnen genutzten Ressourcen, woraus automatisch ein gewisser Grad an Sicherheit resultiert. Schon bald nach ihrem Start 2013 erzielte die Technologie breite Aufmerksamkeit, sodass inzwischen viele Hersteller wie Amazon, Google, HPE, Microsoft, IBM, Red Hat und andere Docker unterstützen. Mit Docker Hub steht ein Online-Dienst zur Verfügung, der eine Registry für Docker-Images und -Repositories umfasst. Zudem besitzt Docker eine API-Schnittstelle zu den Online-Repository-Systemen GitHub und Bitbucket. Ferner implementiert Docker eine eingebaute Versionsverwaltung mit einer direkten Schnittstelle zu Docker Hub.

Links zum Thema

- Homepage von Neo4j
<http://neo4j.com>
- Homepage der Neo4j-Dokumentation
<http://neo4j.com/docs>
- Demo-Driver and How-to for Neo4j Bolt Protocol
<https://github.com/nigelsmall/bolt-howto>
- Neo4j Bolt-Treiber für Java
<https://github.com/neo4j/neo4j-java-driver>
- Neo4j Bolt-Treiber für JavaScript
<https://github.com/neo4j/neo4j-javascript-driver>
- Neo4j Bolt-Treiber für Python
<https://github.com/neo4j/neo4j-python-driver>
- Homepage von MessagePack
<http://msgpack.org>
- Homepage des Neo4j Solution Partners GraphAware
<http://graphaware.com>
- PHP Driver for Neo4js Binary Protocol: Bolt
<https://github.com/graphaware/neo4j-bolt-php>
- PHP Client for Neo4j with Enterprise Support
<https://github.com/graphaware/neo4j-php-client>
- Neo4j JDBC driver for Neo4j 3.x with BOLT protocol
<https://github.com/neo4j-contrib/neo4j-jdbc>
- Neo4j-Spark-Connector based on Neo4j 3.0s Bolt protocol
<https://github.com/neo4j-contrib/neo4j-spark-connector>
- Homepage von Java Mission Control
<http://oracle.com/missioncontrol>
- The Neo4j Java FlightRecorder Integration
<https://github.com/neo4j-contrib/neo4j-jfr>
- Neo4j Creatures Demo
<https://github.com/jexp/neo4j-procedures-demo>
- Javascript Procedures Demo on top of Neo4j 3.0 Procedures
<https://github.com/jexp/neo4j-javascript-procedures>
- Apache Maven Shade Plug-in
<https://maven.apache.org/plugins/maven-shade-plugin>
- Awesome Procedures for Neo4j 3.x
<https://github.com/neo4j-contrib/neo4j-apoc-procedures>
- Homepage des APOC User Guide
<http://neo4j-contrib.github.io/neo4j-apoc-procedures>
- Neo4j Browser User Interface Guide
<http://neo4j.com/developer/guide-neo4j-browser>
- Zusatzinformationen zu Neo4j Browser Sync
<http://neo4j.com/neo4j-browser-sync>
- Neo4j Browser Sync Terms of Use
<http://neo4j.com/terms/neo4j-browser-sync>
- Download-Site für die Docker Toolbox
<https://www.docker.com/products/docker-toolbox>
- Offizielles Repository von Neo4j auf Docker Hub
https://hub.docker.com/_/neo4j
- Docker Hub Repository mit Experimental Neo4j Docker Images
<https://hub.docker.com/r/neo4j/neo4j-experimental>
- Download-Site für Neo4j Enterprise und Community Edition
<http://neo4j.com/download>
- Webseite mit den Neo4j Release Notes
<http://neo4j.com/release-notes>
- Homepage der aktuellen Cypher Refcard
<http://neo4j.com/docs/stable/cypher-refcard>
- Homepage des Open-Source-Projekts openCypher
www.opencypher.org
- Überblick zu den primitiven Datentypen von Java
<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>
- Zusammenfassung der regulären Ausdrücke für Java
<http://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html>

Erst wenn man diesen letzten Releasestand der Version 2.x.a erreicht hat, greift der automatische Upgrade-Prozess der geplanten Zielversion 3.0.y. Entsprechendes gilt auch für die Migration einer 3.0.x-Version auf die aktuellste Neo4j-Version.

Upgrading von der jeweils konkreten Neo4j-Implementierung abhängig

Zuerst führt man wie beschrieben alle – gegebenenfalls inkrementellen – Updates auf den letzten Releasestand der derzeit installierten Neo4j-Version durch.

Um anschließend explizit über einen Neustart von Neo4j die Strukturen der Datenbank beim Update zu aktualisieren, muss der Parameter `allow_store_upgrade` in der Konfigurati-

onsdatei `neo4j.properties` auf `true` stehen. Danach erstellt man vor dem Upgrade auf die neue Version 3 ein Backup der Datenbank, sodass man im Fehlerfall einen Restore vornehmen kann. Nach diesen Vorbereitungen kann man eine Installation der gewünschten Version 3.0.x durchführen, zunächst jedoch ohne dass man die Datenbank über die Startprozedur aktiviert.

Die Erstellung eines Backups und die Durchführung des Upgrading erfolgt in Abhängigkeit von der konkret vorliegenden Neo4j-Implementierung: Single-Instanz (Community Edition), Enterprise Edition und HA-Cluster (High Availability).

Im Fall einer Single-Instanz nimmt man das Backup der Graphdatenbank direkt auf der Betriebssystemebene vor.

Dann gilt es, die aktuellen Einstellungen zur Konfiguration der zu migrierenden Datenbank zu recherchieren. Diese sind nachgelagert auf die neue Version 3 zu übertragen.

Um sich Arbeit zu sparen, nutzt man das config-migrator-Tool, da viele Einstellungen entfallen und neue hinzugekommen sind. Es empfiehlt sich, die mit einer Version 2.x erstellte alte Graphdatenbank mittels `neo4j-admin` in das Home-Verzeichnis der neuen Graphdatenbank zu importieren.

Dabei muss der Parameter `dbms.allow_format_migration` in der `neo4j.conf`-Datei den Wert `true` einnehmen. Nur dann führt der erstmalige Programmstart der neuen Version 3 ein Upgrade der Datenbankstrukturen auf das aktuelle Release durch. Dabei muss die Datenbank-Datei den korrekten Namen aufweisen, im Bedarfsfall setzt man diesen in der `neo4j.conf`-Datei entsprechend über den Parameter `dbms.active_database`.

Danach sollte man den Wert dieses Konfigurationsparameters aus der `neo4j.conf`-Datei entfernen oder auf `false` setzen. Für die Erstellung des Backups steht für Enterprise Edition und Neo4j-HA-Cluster ein spezielles Neo4j-Backup-Tool zur Verfügung.

Sollte man bereits über ein Backup verfügen, so ergänzt `neo4j-backup` diese Datei inkrementell mit den noch fehlenden oder zu aktualisierenden Daten. Besitzt man noch kein Backup oder zeigt der `-to`-Pfad für die Ablage des Backups auf ein leeres Verzeichnis, so führt der Administrationsbefehl `neo4j-backup` einen vollständigen Backup mit den aktuellen Daten durch.

Dazu muss der zugehörige Parameter `dbms.backup.enabled` in der Konfigurationsdatei `neo4j.conf` standardmäßig aktiviert und die Portadresse (6362) über die Konfigurationseinstellung `dbms.backup.address` korrekt gesetzt sein.

Spezielle Vorgehensweise im Cluster-Betrieb für Master und Slaves

Im Fall eines Neo4j-HA-Clusters gilt es die Konsistenz des Gesamtsystems, das heißt des Master-Knotens und aller zugehörigen Slave-Knoten, sicherzustellen. Dies erreicht man durch konsequente Einhaltung aller Einzelschritte beim Upgrade-Prozess.

Nach erfolgreichem Backup des HA-Clusters stoppt man den Master- sowie alle zugehörigen Slave-Knoten.

Danach entfernt man auf allen Slaves die Datenbank-Dateien und installiert auf der Hardware des Masters sowie allen Slaves die neue gewünschte Neo4j-Version. Anschließend führt man für Master und alle Slaves getrennt die folgenden Upgrade-Schritte aus:

- **Master-Knoten:** Zunächst deaktiviert man den HA-Betrieb mittels des Konfigurationsparameters `dbms.mode=SINGLE` und führt dann das Upgrading wie im Fall eines Single-Instanz-Betriebs inklusive spezifischer Konfiguration durch. Danach schaltet man den Master über den Konfigurationsparameter `dbms.mode=HA` in der `neo4j.conf`-Datei wieder auf den HA-Betrieb um. Für die anschließende Migration des Graphen auf die Slave-Knoten erstellt man mittels `neo4j-backup` ein vollständiges Sicherheits-Backup der aktuellen Datenbank.

Evolutionäre Entwicklung der Zugriffsschnittstellen

Neo Technology bereitete den Einsatz der Graphdatenbank für den Einsatz im Massenbetrieb kontinuierlich vor.

In der frühen Entwicklungszeit von Neo4j besaß das DBMS für die Anwendungsentwicklung lediglich ein Embedded Java-API; Programme griffen ausschließlich über die in Neo4j implementierte Java-Schnittstelle (Core-API) auf die Graphdatenbank zu.

Mit Version 1 stellte der Hersteller für die Zugriffe auf die Datenbank eine REST-Schnittstelle (Representational State Transfer) bereit. Damit besaß Neo4j erstmals mittels des HTTP(S)-Protokolls eine einheitliche Schnittstelle für die Programmierung einer Graphanwendung.

Als wichtiger Schritt bei der Weiterentwicklung von Neo4j stellte Neo Technology in der Version 2 eine eigene Abfragesprache für Graphen, Cypher Query Language (CQL) genannt, vor. Cypher versetzt Entwickler, aber auch Endbenutzer in die Lage, direkt mit der Graphdatenbank zu kommunizieren.

Mit dem neuen BOLT-Protokoll erweitert der Hersteller über neue offiziell unterstützte Sprachschnittstellen den Kreis der Entwickler und bietet eine optimale Performance für Graphen gigantischer Größe. BOLT eignet sich speziell für die schnelle Übertragung mehrerer Bestandteile einer Graphdatenbank.

- **Alle Slave-Knoten:** Für jeden Slave-Knoten überführt man die jeweils spezifischen Einstellungen der DBMS-Konfiguration auf die neu installierte neue Neo4j-Version. Eine eventuell vorhandene Security-Konfiguration muss manuell vom Master auf alle Slaves übertragen werden. Anschließend prüft man, ob alle Slaves den erforderlichen Datenbanknamen des Masters besitzen. Gegebenenfalls definiert man diesen in der `neo4j.conf`-Datei über den Wert des Eintrags `dbms.active_database`.

Sind die oben genannten Vorkehrungen getroffen, so führt man einen Restart des gesamten Cluster-Systems durch. Hierzu startet man die Master-Instanz und sequenziell nachgelagert jeden einzelnen Slave-Knoten. Sobald ein Slave sich in den Cluster-Verbund einklinkt, findet zuerst eine Synchronisation seiner lokalen Datenbank mit der Master-Datenbank statt. Nach dieser Operation besitzt der komplette Cluster-Verbund den aktuellen Datenbestand und der HA-Betrieb läuft mit der neuen und aktuellen Neo4j-Version. ■



Frank Simon

arbeitet in der Software-Entwicklung mit den aktuellen Arbeitsgebieten Entwicklung, Programmierung, Test und Debugging von Cloud-, Rich-Internet-, Mobile- und Webanwendungen inklusive deren System-Management.

web_mobile_developers@gmx.eu

WEB APPLICATION SECURITY SCANNING MIT NIKTO

Potenzielle Ziele

Webbasierte Applikationen bilden vielfach das Herzstück der IT-Infrastruktur eines Unternehmens und müssen daher optimal gesichert werden.

Webbasierte Applikationen bilden in vielen Unternehmen das Herzstück der IT-Infrastruktur. Der Trend geht in fast allen Bereichen zu Webapplikationen. Ob es sich dabei um ein internes Content-, CRM-, Dokumentenmanagement- oder Workgroup-System handelt, ist zunächst zweitrangig. Sie alle sind für potenzielle Angreifer interessante Ziele.

Das ist nicht weiter verwunderlich, denn in webbasierten Umgebungen wie beispielsweise einem Magento-Shop sind durchaus interessante Informationen zu finden: Kontakt-, Konten- und Kreditkartendaten et cetera. Aufgrund ihrer komplexen Architektur gelten sie als besonders anfällig für Attacken. Ein umfassender Schutz ist mehrstufig: Mit einer Web Application Firewall schützen Sie Ihre Webapplikation vor Attacken von außen; mit einem Web Application Security Scanner prüfen Sie Ihre Anwendung auf mögliche Schwachstellen und Sicherheitslücken.

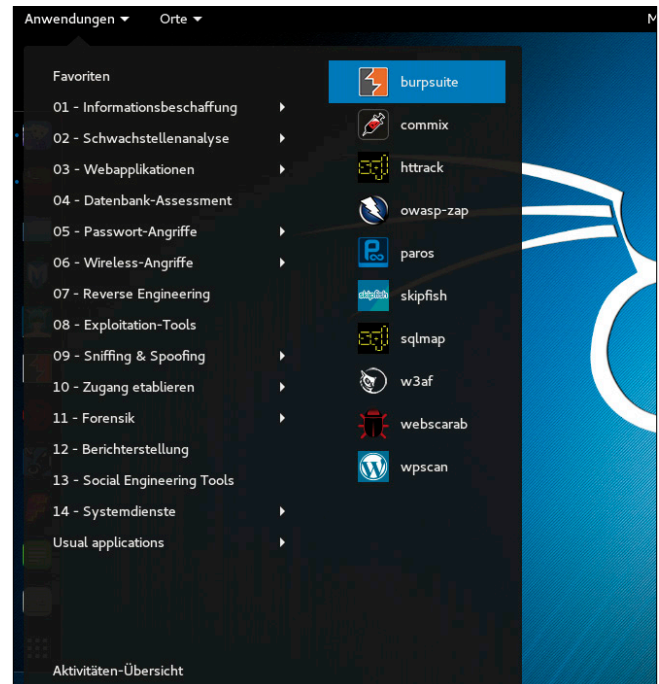
Webbasierte Umgebungen wie Magento und Co. sind komplexe Gebilde, die auf einer nicht weniger komplexen Umgebung (Apache, MySQL, PHP) basieren. Damit ergeben sich für einen potenziellen Angreifer verschiedenste Angriffspunkte.

Doch bei der Komplexität von Webapplikationen ist es alles andere als einfach, Schwachstellen aufzuspüren. Hier kommen Sie mit Web Application Security Scannern weiter. Sie helfen, Sicherheitslücken und Schwachstellen in gängigen Webanwendungen aufzuspüren. Mit dem gewonnenen Know-how kann man sich dann an das Schließen der Lücken und das Härten der Umgebung machen.

Penetration Testing

Das Testen von Applikationen und Infrastrukturkomponenten wird auch als Penetration Testing bezeichnet. Hat man die Notwendigkeit von eingehenden Sicherheitstests erkannt, folgt unmittelbar die Frage nach einem geeigneten Werkzeug. Für Penetration-Tester gilt Kali Linux, eine spezielle Linux-Distribution mit Hunderten Sicherheitswerkzeugen, als Standardumgebung.

In Kali Linux sind auch verschiedene Spezialisten für die Analyse von Webapplikationen enthalten. Wenn Sie in Kali Linux einen Blick in das Menü *Anwendungen*, *Kali Linux*, *Webapplikationen* und *Webschwachstellen-Scanner* werfen, werden Sie nicht schlecht staunen: Dort finden Sie zehn Tools für das Aufdecken von Schwachstellen in Webapplikationen. Als die besten Web Application Security Scanner gelten die Burp Suite (<http://portswigger.net/burp>), w3af (<http://w3af.org>) und Nikto2 (<https://cirt.net/Nikto2>).



In Kali Linux sind jede Menge Tools für die Prüfung von Webapplikationen enthalten (Bild 1)

Unter Penetration-Testern erfreut sich Nikto großer Beliebtheit, weil dieser Scanner umfangreiche Tests gegen Webserver erlaubt. Laut Angaben der Entwickler kann Nikto über 6700 potenzielle Schwachstellen und mehr als 1200 veraltete Server-Versionen ermitteln. Hinzu kommen noch einmal 270 versionsspezifische Sicherheitsprobleme.

Der Security-Scanner kann auch sicherheitsrelevante Konfigurationsprobleme wie die Existenz von mehreren Index-Dateien und ungünstige HTTP-Server-Optionen identifizieren. Nikto kann in der Regel die installierten Webserver-Komponenten zuverlässig ermitteln sowie Objekte und Plugins erkennen, die häufig aktualisiert werden.

Das Tool ist so konzipiert, dass es möglichst flott den zu untersuchenden Webserver unter die Lupe nimmt und die Ergebnisse in eine Log-Datei schreibt beziehungsweise die Daten an ein Intrusion Prevention System (IPS) oder an ein Intrusion Detection System (IDS) übergibt.

Nicht jede Prüfung ermittelt notwendigerweise eine Schwachstelle oder Sicherheitslücke. Verschiedene Tests dienen schlicht dazu, den Webmaster beziehungsweise Administrator auf bestimmten Zustände hinzuweisen. Entspre-

chende Hinweise sind als reine Info-only-Bemerkungen zu verstehen. Unabhängig davon kann es natürlich auch vorkommen, dass Nikto einem für den Scanner noch unbekannten Problem begegnet und dies als informellen Hinweis in den Logfiles notiert. Im Zweifel müssen entsprechende Einträge manuell von Verantwortlichen geprüft werden.

Nikto hat weitere Besonderheiten zu bieten. Der Scanner schreibt seine Ergebnisse standardmäßig in eine Textdatei, kann diese aber auch nach XML, HTML, NBE und CSV exportieren. Dank einer komfortablen Template-Engine können Sie benutzerdefinierte Berichte erstellen.

Das Scannen beschleunigt sich deutlich, indem Sie mehrere Ports gleichzeitig untersuchen. Dabei kann auch die Ausgabe einer Nmap-Installation verwendet werden. Nikto kann die auf einem Webserver installierte Software über Header, Favicons und spezifische Dateien ermitteln. Eine Authentifizierung kann über ein Standard-Login und NTLM erfolgen. Der Scanner kann auch Sub-Domains erraten. Um den Scanvorgang zu beschleunigen, kann Nikto ganze Verwundbarkeitsklassen ausschließen. Sogar eine Behandlung von False Positives ist möglich, um die Zahl der gefundenen Verwundbarkeiten auf tatsächliche Schwachstellen zu reduzieren. Tests können gespeichert und bei Bedarf zu bestimmten Zeiten fortgeführt werden.

Nikto ist eine Perl-basierte Applikation, die auf jedem beliebigen Betriebssystem eingesetzt werden kann, das über einen Perl-Interpreter verfügt. Wenn Sie Kali Linux verwenden wollen, steht dort Nikto unmittelbar zur Verfügung (Bild 1).

Erste Schritte mit dem Security Scanner

Auch wenn es manchen Anwendern nicht unbedingt zeitgemäß erscheint, so gibt es doch nach wie vor viele Sicherheitspezialisten, die über die Konsole ausgeführt werden, beispielsweise der Netzwerkscanner Nmap und das Exploit-Framework Metasploit. Auch Nikto ist ein Konsolenprogramm. Die ersten Gehversuche führen Sie am besten an einem lokalen Testsystem aus, das Sie schnell und einfach mit XAMPP aufsetzen können.

```

root@kali: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
root@kali:~# nikto -h 192.168.2.1 -T 58
- Nikto v2.1.6
-----
+ Target IP:      192.168.2.1
+ Target Hostname: 192.168.2.1
+ Target Port:    80
+ Start Time:     2016-06-13 11:30:51 (GMT2)
-----
+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user
  agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent
  to render the content of the site in a different fashion to the MIME type
+ Root page / redirects to: http://192.168.2.1/html/login/index.html
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading
  HTTP response
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading
  HTTP response
+ Scan terminated: 21 error(s) and 3 item(s) reported on remote host
+ End Time:       2016-06-13 11:30:52 (GMT2) (1 seconds)
-----
+ 1 host(s) tested
root@kali:~#

```

Nikto ist ebenfalls in Kali Linux installiert und liefert schnell und zuverlässig erste sicherheitsrelevante Informationen über das Zielsystem (Bild 2)

```

# should
# send updates because it makes the data better for everyone ;) *NO* server specific
information
# such as IP or name is sent, just the relevant version information.
# UPDATES=yes - ask before each submission if it should send
# UPDATES=no - don't ask, don't send
# UPDATES=auto - automatically attempt submission *without prompting*
UPDATES=yes

# Warning if MAX_WARN OK or MOVED responses are retrieved
MAX_WARN=20

# Prompt... if set to 'no' you'll never be asked for anything. Good for automation.
# PROMPTS=no

# cirt.net ; set the IP so that updates can work without name resolution -- just in case
CIRT=107.170.99.251

# Proxy settings -- still must be enabled by -useproxy
#PROXYHOST=127.0.0.1
#PROXYPORT=8080
#PROXYUSER=proxyuserid
#PROXYPASS=proxypassword

# Cookies: send cookies with all requests
# Multiple can be set by separating with a semi-colon, e.g.:
# "cookie1="cookie value"; cookie2="cookie val"
#STATIC-COOKIE=

# The below allows you to vary which HTTP methods are used to check whether an HTTP(s) server
# is running. Some web servers, such as the autopsy web server do not implement the HEAD method
CHECKMETHODS=HEAD GET

```

Sicherheitsprüfungen können permanent über einen Proxy Server ausgeführt werden (Bild 3)

Um einen lokalen Webserver mit beispielsweise der IP-Adresse 192.168.2.1 auf Port 80 zu prüfen, führen Sie folgenden Befehl aus (Bild 2):

```
nikto -h 192.168.2.1
```

Die Option `-h` ist die Host-Option. Erfolgt keine explizite Angabe des Ports, geht Nikto davon aus, dass Sie sich für den Standard-Port 80 interessieren.

Um einem spezifischen Port zu prüfen, verwenden Sie die Option `-p`, die Port-Option. Um beim Testsystem Port 443 zu prüfen, führen Sie folgenden Befehl aus:

```
nikto -h 192.168.2.1 -p 443
```

Sie können auch Host-, Port und Protokollangaben mit einer vollständigen URL-Syntax verwenden:

```
nikto -h https://192.168.2.1:443/
```

Es besteht übrigens keine Notwendigkeit zu spezifizieren, dass Port 443 ein SSL-Port ist. Nikto verwendet zunächst immer HTTP. Schlägt das fehl, erfolgt die Prüfung mit HTTPS. Allerdings können Sie die Prüfung mit der Option `-ssl` beschleunigen:

```
nikto -h 192.168.2.1 -p 443 -ssl
```

In Verbindung mit dem Parameter `-mutate` können Sie auch komplexe Tests durchführen. Sie können den Scanner sogar für die Durchführung von Brute-Force-Attacks verwenden:

```
nikto -h 192.168.2.1 -mutate 3 -mutate-option
user-list.txt
```

Um mehrere Ports in einem Scan-Vorgang zu prüfen, spezifizieren Sie die Ports in der Port-Liste. Dabei können Sie ganze Port-Bereiche (zum Beispiel `80-88`) oder aber eine kom-

maseparierte Liste (zum Beispiel *80,88,443*) verwenden. Hier ein einfaches Beispiel für das Testen einer Liste:

```
nikto -h 192.168.2.1 -p 80,88,443
```

Wenn Sie mehrere Hosts in einem Durchgang prüfen wollen, legen Sie hierzu eine Host-Datei an, in der die Hosts aufgeführt werden – und zwar immer mit einem Host pro Zeile und der Port-Angabe am Ende der Zeile. Eine gültige Hostliste kann unterschiedliche Host-Spezifikationen enthalten:

```
192.168.2.1:80
http://192.168.2.2:8080/
192.168.2.3
```

Sie können auch die Ausgabe des Netzwerkanalyse-Spezialisten Nmap verwenden. Hier ein Beispiel:

```
nmap -p80 192.168.2.0/24 -oG - | nikto -h
```

Nikto bietet neben den Standardtests eine Vielzahl von erweiterten Funktionen. Erfolgt der Zugriff auf ein zu testendes System nur über einen Proxy Server, so ist auch hier eine Prüfung mit Nikto möglich. Der Scanner kennt zwei Möglichkeiten: Entweder verwendet man die Konsole, oder aber man spezifiziert den Proxy Server in der Nikto-Konfigurationsdatei *nikto.conf*. Zweites ist sinnvoll, wenn die Verbindungsaufnahme permanent über einen Proxy Server erfolgt. In diesem Fall muss die Variable *PROXY* gesetzt werden (Bild 3).

Auf der Konsolenebene verwenden Sie die Option *-use-proxy*. Hierzu ein Beispiel:

```
nikto -h localhost -useproxy http://localhost:8080/
```

Nikto greift bei seinen Tests auf verschiedene Plug-ins und Datenbanken zurück. Damit Ihnen keine relevanten Sicherheitslücken und Schwachstellen entgehen, sollten Sie vor dem Einsatz ein Update dieser Daten durchführen. Auch das ist denkbar einfach:

```
nikto -update
```

Sollte das Herunterladen von Aktualisierungen erforderlich sein, können Sie das auf der Konsole verfolgen:

```
+ Retrieving 'ikto_core.plugin'
+ Retrieving 'CHANGES.txt'
```

Eine Besonderheit von Nikto: Sie können während der Tests mit dem Programm interagieren. Dazu betätigen Sie während eines aktiven Scan-Vorgangs eine der folgenden Tasten:

- Leertaste – Gibt den aktuellen Scan-Status aus,
- v – Schaltet den geschwätzigen Modus ein und aus,
- d – Schalte den Debug-Modus ein und aus,
- e – Schaltet die Fehlerberichte ein und aus,
- p – Schaltet die Fortschrittanzeige ein und aus,
- c – Schaltet Cookie-Informationen ein und aus,

- a – Blendet das Authentifizierungsfenster ein und aus,
- q – Beendet die Ausführung von Nikto,
- N – Wechselt zum nächsten Host,
- P – Pausiert die Ausführung.

Wenn Sie sich für bestimmte Schwachstellen und Sicherheitslücken des Zielsystems interessieren, können Sie die Tests zielgenau ausführen. Manchmal interessiert man sich nur für bestimmte Dateien, das andere Mal nur für Cross-Site-Scripting-Anfälligkeiten. Das Testverhalten können Sie mit der Option *-T* steuern.

Um Tests ausschließen, kombinieren Sie die Option *T* mit einem x. Dann werden alle Tests nach dem x negiert. Das ist beispielsweise nach der Ausführung von verschiedenen Exploits sinnvoll. Ihnen stehen dabei folgende Tuning-Optionen zur Verfügung:

- 0 – dient dem Upload einer Datei auf das Zielsystem,
- 1 – identifiziert interessante Dateien oder Ereignisse,
- 2 – identifiziert Dateien mit ungünstigen Konfigurationen,
- 3 – identifiziert Informationen über das Zielsystem,
- 4 – führt Cross-Site Scripting oder Content Injection aus,
- 5 – lädt Dateien aus dem Web-Root-Verzeichnis herunter,
- 6 – führt eine Denial-of-Service-Attacke gegen das Ziel aus,
- 7 – ruft serverweit Informationen über Dateien ab,
- 8 – erlaubt die Remote-Ausführung von Befehlen,
- 9 – führt eine SQL-Injection-Attacke aus,
- a – umgeht den Authentifizierungsschutz,
- b – identifiziert auf dem Server installierte Software,
- c – erlaubt das Einbringen von Sourcecode,
- x – schließt bestimmte Tuning-Option aus.

Ab Version 2.1.2 ist bei der Ausführung von Nikto die Auswahl von Plug-ins möglich. Auf diesem Weg lassen sich sehr spezielle Tests durchführen. Dabei kommt die Plug-in-Option zum Einsatz. Die Liste der verfügbaren Plug-ins rufen Sie mit

```
root@kali:~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
root@kali:~# nikto -list-plugins
Plugin: report_sqlg
Generic SQL reports - Produces SQL inserts into a generic database.
Written by Sullo, Copyright (C) 2013 CIRI Inc.

Plugin: siebel
Siebel Checks - Performs a set of checks against an installed Siebel application
Written by Tautology, Copyright (C) 2011 CIRI Inc.
Options:
  applications: List of applications
  languages: List of Languages
  application: Application to attack
  enumerate: Flag to indicate whether we shall attempt to enumerate known apps

Plugin: tests
Nikto Tests - Test host with the standard Nikto tests
Written by Sullo, Tautology, Copyright (C) 2008 CIRI Inc.
Options:
  passfiles: Flag to indicate whether to check for common password files
  report: Report a status after the passed number of tests
  all: Flag to indicate whether to check all files with all directories
  tids: A range of testids that will only be run

Plugin: report_xml
Report as XML - Produces an XML report.
Written by Sullo/Jabra, Copyright (C) 2008 CIRI Inc.

Plugin: spacchusers
Apache Users - Checks whether we can enumerate usernames directly from the web server
Written by Javier Fernandez-Sanguinoi Pena, Copyright (C) 2008 CIRI Inc.
Options:
  home: Look for -user to enumerate
  dictionary: Filename for a dictionary file of users
  size: Maximum size of username if bruteforcing
  enumerate: Flag to indicate whether to attempt to enumerate users
```

Der Plug-in-Liste können Sie entnehmen, welche spezifischen Tests mit Nikto möglich sind und wie Sie diese Erweiterungen verwenden (Bild 4)

nikto -list-plugins ab (Bild 4). Dabei können auch Makros verwendet werden.

Die Steuerung eines Programms aus der Konsole heraus ist nicht jedermanns Sache. Dafür gibt es mehrere Gründe. So ist die Ausführung fehleranfällig, weil man mit den verfügbaren Optionen und Parametern und deren exakter Verwendung vertraut sein muss. Deutlich einfacher ist die Nutzung über eine klassische Programmoberfläche oder ein Webinterface.

Ein ausgesprochen leistungsfähiger Web Applikation Security Scanner, der über eine Webschnittstelle gesteuert wird, ist die Burp Suite. Der Scanner ist für Windows und Linux gleichermaßen verfügbar und steht in einer Free Edition kostenlos bereit (Bild 5).

In Kali Linux ist ebenfalls die Free Edition integriert. Wenn Sie die Burp Suite unter Windows einsetzen wollen, ist auch das einfach: Laden Sie sich die JAR-Datei herunter und führen Sie diese aus. Wer Gefallen an den Möglichkeiten der Burp Suite findet und noch mehr Funktionalität wünscht, kann für circa 300 US-Dollar ein Jahresabo erwerben, das weitere Funktionen bietet, etwa eine komfortable Scanfunktion, die aktives und passives Scannen unterstützt, eine leistungsfähige Suche, eine Content-Analyse und Zeitsteuerungen. Für den Einstieg genügt die Free Edition in jedem Fall.

Die Burp Suite ist eine integrierte Plattform für die Durchführung von Sicherheitstests an Webapplikationen. Sie bietet Ihnen die volle Kontrolle für die auszuführenden Tests und Aktionen. Die Burp Suite bietet außerdem tiefgehende Analysen der Ergebnisse.

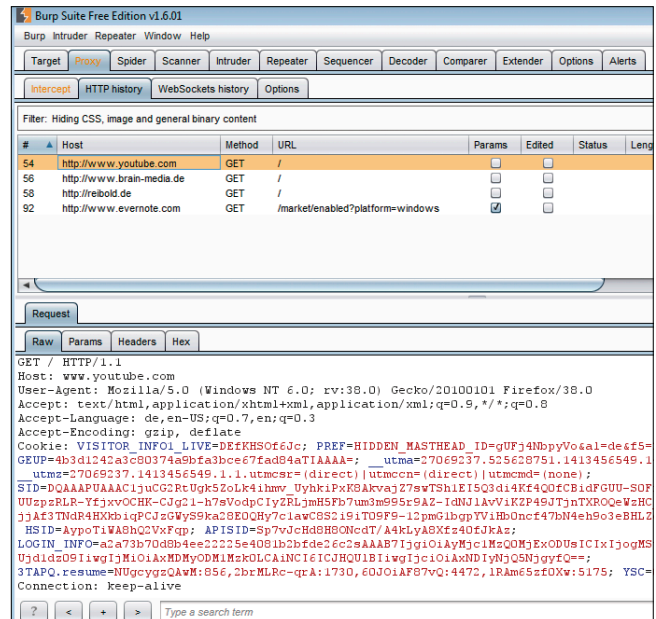
Die Suite ist auf das Zusammenspiel mit einem Webbrowser hin konzipiert, über den beispielsweise Eingaben an eine Webapplikation übermittelt und dann von der Security Suite analysiert werden. Um Tests mit der Burp Suite durchführen zu können, müssen Sie die Proxy-Einstellungen Ihres Browsers bearbeiten. Tragen Sie dort den URL 127.0.0.1:8080 ein.

Stellen Sie außerdem in den Burp-Einstellungen sicher, dass der Proxy Listener aktiviert ist. Sie finden dessen Einstellungen auf der Registerkarte *Proxy* im Unterregister *Options*. Stellen Sie unter *Proxy Listener* sicher, dass obige Adresse verwendet wird und aktiviert ist.

Sie müssen eine weitere Anpassung vornehmen, bevor Sie mit Ihren ersten Checks loslegen können: Öffnen Sie die Registerkarte *Proxy Intercept*. Stellen Sie dort sicher, dass die *Interception*-Funktion aktiviert ist.

Öffnen Sie nun Ihren Browser und fordern Sie ein beliebiges Dokument an. Jeder Request Ihres Browsers wird nun auf der Registerkarte *Interception* angezeigt. Burp verfügt außerdem über eine Proxy-History, die alle Requests und Responses aufzeichnet. Den Verlauf finden Sie auf der *History*-Registerkarte.

Wenn Sie eine Website ansteuern, erzeugt Burp eine Sitemap einer Website. Die Ergebnisse dieses Vorgangs sind auf der Registerkarte *Target* auf dem Unterregister *Site Map* verfügbar. In dieser Sitemap sind alle URLs enthalten, die Sie besucht haben, und solche, die Burp darüber hinaus aufdecken konnte. Alle Objekte, die Sie angefordert haben, werden schwarz markiert, alle anderen grau.



Grafische Oberfläche: Ein Blick auf die Burp Suite Free Edition unter Windows (Bild 5)

Die Burp Suite ist so konzipiert, dass Sie alle Aktionen selbst steuern können. Sie können dabei insbesondere alle HTTP-Requests mit den Burp-Tools an eine webbasierte Applikation übergeben. Für bestimmte Aufgaben stehen Ihnen spezifische Funktionen und Tools zur Verfügung, die über die gleichnamigen Registerkarten verfügbar sind:

- Spider bewegt sich automatisch durch eine Applikation, um deren Struktur und Funktionalität zu ermitteln.
- Scanner durchsucht HTTP-Requests nach bekannten Verwundbarkeiten.
- Intruder führt benutzerdefinierte Attacken aus.
- Repeater wiederholt manuell modifizierte Requests immer und immer wieder.
- Sequencer analysiert die Qualität der Session-Zeichen.
- Decoder dient dem Decodieren und Codieren von Anwendungsdaten.
- Comparer dient dem visuellen Vergleich von Anwendungsdaten.

Sie können dabei die Tools kombinieren. Wenn Sie erste Erfahrungen mit einem Web Application Security Scanner sammeln, ist auch das kein Problem, denn Sie können die Burp Suite als Point-and-Click-Scanner verwenden. ■



Holger Reibold

arbeitet seit über 20 Jahren als IT-Journalist und Autor mit den Schwerpunkten Internet, Open Source und Webtechnologien. Außerdem arbeitet er als E-Commerce-Manager.

www.reibold.de



QUALITÄTSSICHERUNG AS A SERVICE

Software testen in der Cloud

Cloud-Plattformen können Kosten senken und die Entwicklung beschleunigen.

Software-Entwickler stehen unter einem erheblichen Druck. Sie müssen Applikationen und mobile Apps schneller zur Marktreife bringen und häufiger aktualisieren als je zuvor. Treten dabei Fehler oder gar Sicherheitslücken auf, können die Folgen gravierend sein. Viele suchen deshalb nach Wegen, Software zeitsparend, flexibel, kostengünstig, gründlich und – wo immer möglich – automatisiert testen zu können.

»Mit den wachsenden Anforderungen des Marktes, die es verlangen, eine Vielzahl von Browsern und Geräten zu unterstützen, steigt auch das Interesse vieler Unternehmen an automatisierten und skalierbaren Testing-Lösungen«, sagt Claudia Wembacher, Produkt Marketing Manager für Cloud Application Development bei Microsoft.

Langfristig werde es sich kein Unternehmen mehr leisten können, eine eigene Plattform zu entwickeln, die diese Anforderungen erfüllt, ergänzt Heiko Specht, Regional Sales Manager DACH beim Cloud-Testing-Pionier SOASTA: »Software as a Service für Test-Herausforderungen wird deshalb in Zukunft wesentlich stärker nachgefragt werden.« Auch Christian Schütz, Business Development Manager Application Delivery Management and Enterprise Security bei HPE

Software Deutschland, sieht eindeutig den Trend hin zu Testing aus der Cloud.

Der Markt für Testing als Service sei bislang klein, expandiere aber rasch, sagt Matthias Zacher, Senior Consultant beim Marktforschungsunternehmen IDC Central Europe. Noch liegt der Hauptanteil in Amerika. Dort wurde 2015 bereits ein Marktvolumen von rund 400 Millionen Dollar erreicht, in EMEA waren es nur 115 Millionen Dollar. Mit einem prognostizierten durchschnittlichen jährlichen Wachstum von knapp 40 Prozent in den kommenden drei Jahren hat der EMEA-Raum aber gute Chancen aufzuschließen, da der Gesamtmarkt laut IDC »nur« knapp 30 Prozent wachsen und bis 2019 einen Umfang von 1,5 Milliarden Dollar erreichen soll.

Die Marktforscherkollegen von Forrester Research sehen Testing as a Service ebenfalls erst am Anfang. Derzeit werde die Cloud eher für Entwicklungs- als für Testzwecke eingesetzt, schreibt Diego Lo Giudice, Vice President und Principal Analyst Application Development & Delivery, im Report »Improve Quality And Speed With Application Testing In The Cloud«. Während fast 40 Prozent der von Giudice und seinem Team befragten Entwickler die Cloud für die Applikations-

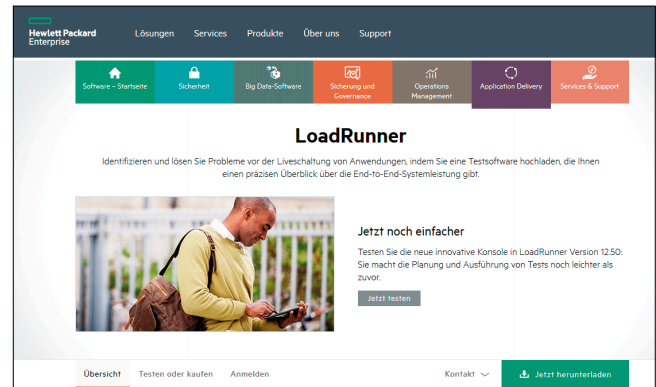
Integration und für firmeneigene Web-Apps nutzen, sind es bei Test und Qualitätssicherung erst ein Drittel. Forrester weist für den Testmarkt keine gesonderten Zahlen aus, geht aber davon aus, dass die Akzeptanz von Cloud-Diensten für Applikationsentwicklung und -auslieferung insgesamt in den kommenden zwei Jahren um rund 10 Prozent steigen wird.

Die Vorteile des Cloud Testing

Viele Argumente für Software Testing als Service gelten allgemein für das Cloud Computing. »Die Kunden bewegen sich von CAPEX hin zu OPEX, um die Anfangsinvestitionen zu reduzieren. Zusätzlich verringern sie damit das Risiko einer Fehlinvestition und bleiben in den Lösungen flexibler. Diese Entwicklung unterstützt das Cloud Testing«, sagt Annette Gemeinder, Manager Enterprise Testing Central & Eastern Europe bei CSC. Der IT-Dienstleister bietet mit Partnern wie HPE, Perfecto und BlazeMeter Funktions- und Performance-Tests, aber auch Testautomatisierung sowie Tests für mobile Anwendungen als Service an.

Die Schonung von Kapital und Liquidität ist jedoch nur ein Aspekt, der für Software Testing als Service spricht – und nach Ansicht von Forrester-Analyst Giudice nicht einmal der entscheidende: »Kosteneinsparungen durch schnelle, korrekte und qualitativ bessere Tests sind relevanter und für die Wertschöpfung wichtiger«, sagt er. Diese Anforderungen ließen sich in einer Cloud-Umgebung besser realisieren als im eigenen Rechenzentrum.

Ein weiterer Vorteil besteht nach Ansicht von Georg Hansbauer, Gründer und Geschäftsführer der Testbirds GmbH,



HPE bietet ein umfangreiches Portfolio an Test-Services (Bild 1)

darin, dass man sich bei Testing als Service nicht selbst um die Wartung und Aktualisierung der Testwerkzeuge kümmern muss: »Gerade in einem so schnelllebigem Markt wie der Software-Entwicklung ist es essenziell, die Tools kontinuierlich weiterzuentwickeln, um den Anschluss nicht zu verpassen«, sagt er. »Nicht zuletzt vereinfacht sich das Onboarding neuer Mitarbeiter, vor allem an dezentralen Standorten«, ergänzt HPE-Manager Schütz.

Besonders gut geeignet sei die Cloud für Performance- und Lasttests, bei denen man schnell, aber nur für kurze Zeit sehr viel Rechenleistung benötigt, sagt Karsten Kempe, ALM-Architekt (ALM, Application Lifecycle Management) und Dev-Ops-Spezialist bei Microsoft: »In der Cloud kann man diese zusätzliche Leistung ohne großen Aufwand mehr oder weniger auf Knopfdruck beziehen, den Test durchführen und dann auf ein normales Niveau zurückkehren.«

Manche Fragen lassen sich sogar nur mit Hilfe der Cloud beantworten. Welchen Einfluss beispielsweise große Entfernungen über Kontinente hinweg auf Antwortzeiten und Performance einer Webapplikation haben, kann man ohne Einsatz eines Cloud-basierten Services nur mit einem kaum vertretbaren Aufwand beantworten. »Man kann das Internet nicht in einem Labor simulieren«, sagt SOASTA-Manager Specht.

Vor allem Entwickler mobiler Applikationen stehen vor der Herausforderung, verschiedenste Gerätetypen, Bildschirmgrößen, Hardware-Plattformen mit unterschiedlich leistungsfähigen Prozessoren und Betriebssystemversionen bedienen zu müssen. Für sie sind Device-Farmen oder Cross-Plattform-Tests ideal. »Welche Firma möchte sich schon Tausende Smartphones oder Tablets anschaffen, um auf diesen Geräten manuelle Tests durchzuführen, wenn es dafür Cloud-Dienste gibt, die genau dafür ausgelegt sind«, beschreibt Microsoft-Mitarbeiter Kempe die Vorteile.

Einer der wichtigsten Gründe für Cloud Testing ist nach Ansicht von Forrester-Analyst Giudice die Möglichkeit, standardisierte Testumgebungen zur Verfügung stellen zu können, wie dies beispielsweise der US-amerikanische Public-Cloud-Provider Skytap tut. Die Services erlauben es Entwicklern und Testern unter anderem, selbst Testumgebungen aufzusetzen, diese zu klonen oder als Vorlage für andere Tests zu verwenden. Bei Nichtbenutzung schaltet sich das System automatisch ab, sodass keine Kosten anfallen. ►

Was lässt sich in der Cloud testen?

Es gibt fast nichts, was sich an Software testen lässt, das nicht auch als Service bezogen werden kann. Hier einige Beispiele:

- **Funktion:** Erfüllt die Software die festgelegten Spezifikationen? Hat sie Programmierfehler, die eine Verwendung verhindern, erschweren oder die zu Sicherheitslücken führen?
- **Bedienbarkeit:** Ist die Bedienoberfläche verständlich und ergonomisch? Erreicht ein Anwender schnell, einfach und ohne Umwege sein Ziel?
- **Latenz:** Wie lange sind die Zugriffszeiten bei – womöglich weltweit verteilten – Anfragen?
- **Skalierbarkeit:** Wie reagiert die Applikation oder Webseite auf viele gleichzeitige Zugriffe? Gibt es einen kritischen Punkt, ab dem sie nicht mehr stabil läuft?
- **Plattformunabhängigkeit:** Arbeitet die Software auf allen spezifizierten Betriebssystemen und -versionen, Browsern, Endgeräten, Hardware-Plattformen oder Mobilsystemen?
- **Sicherheit:** Ist die Applikation oder Webseite gegen unautorisierte Zugriffe ausreichend geschützt? Werden sensible Daten in allen drei Stadien (Entstehung, Transfer, Speicherung) verschlüsselt? Sind Nutzerkonten ausreichend gegeneinander isoliert?

Testbirds-Geschäftsführer Georg Hansbauer sieht ebenfalls sehr großes Potenzial in der Teststandardisierung und -automatisierung: »Mit den richtigen Schnittstellen lassen sich automatisierte Tests schnell und einfach in der Cloud umsetzen.«

Nachteile vom Testen in der Cloud

Vieles spricht für das Testen in der Cloud, man muss allerdings vor der Auslagerung seiner Software-Qualitätssicherung einiges bedenken, um unliebsame Überraschungen zu vermeiden. Vor allem der Aufwand für die Umstellung ist nicht zu unterschätzen. »Die initialen Setup-Kosten für die Migration der Tests in die Cloud sind relativ hoch«, warnt Annette Gemeinder von CSC, »die Testfälle müssen für eine Durchführung in der Cloud-Umgebung angepasst werden«.

Oft lässt sich die bisherige Testumgebung nicht eins zu eins in der Cloud abbilden. Dann muss der gesamte Testprozess auf den Prüfstand – angefangen von der Release-Planung mit den jeweiligen Testphasen und Testplänen, Personaleinsatz und Zeitaufwand über Tools und Schnittstellen bis hin zum Reporting und der Interpretation von Testergebnissen. Womöglich sind Testfälle, Datenbanken, Bibliotheken oder Abhängigkeiten in der Cloud nicht ohne Weiteres umzusetzen und müssen umgebaut werden.

Dieser Aufwand lohnt sich nicht in jedem Fall, sagt Heiko Specht von SOASTA: »Es gibt Tests, die so spezifisch sind, dass eine Implementierung in Test-Tools eher mehr Zeit beansprucht, als würden diese manuell getestet.«

Ein weiterer Punkt, der sich laut Georg Hansbauer nicht ohne Weiteres in die Cloud auslagern lässt, ist das Testmanagement: »Die manuelle Koordination vor Ort ist ein wichtiger Bestandteil des kompletten Prozesses«, sagt der Testbirds-Chef.

Bei der Migration können Prozessmanagement-Methoden wie SMART-T helfen, eine speziell für das Testen adaptierte Version des SMART-Prozesses (SOA Migration, Adoption, and Reuse Technique), der, wie der Name schon andeutet, Unternehmen bei der Überführung von Applikationen in ei-

Überblick: Cloud-basierte Testplattformen

- Amazon Web Services
www.aws.amazon.com/de
- BlazeMeter
www.blazemeter.com
- CSC
www.csc.com/de/testing_services
- Hewlett Packard Enterprise
saas.hpe.com/de-de/home
- Microsoft
www.azure.com
- Neotys
www.neotys.de
- RadView
www.radview.com/de
- SOASTA
www.soasta.de
- Testbirds
www.testbirds.com/technologies/cloudtechnologies

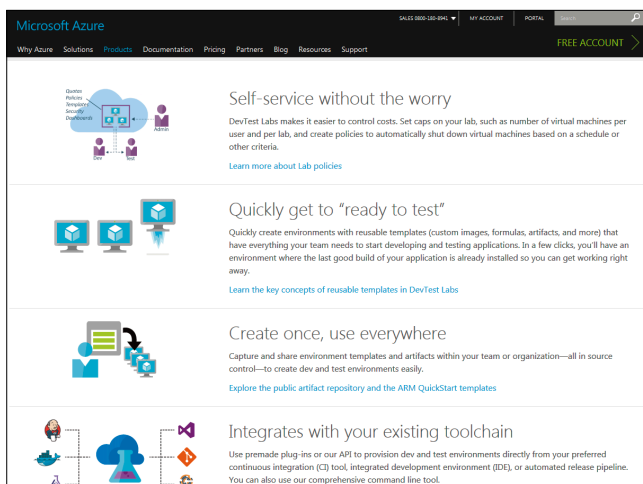
ne serviceorientierte Architektur unterstützen soll. (Detaillierte Angaben zu SMART-T finden sich in dem Buch »Software Testing in the Cloud« von Scott Tilley und Tauhida Parveen, erschienen im Springer Verlag.)

Bei Latenz- und Performance-Tests, aber auch bei Funktionstests kommen Unwägbarkeiten hinzu, die durch den Transfer in eine weitgehend unbekannte, vom Unternehmensstandort womöglich weit entfernte Umgebung entstehen. Welchen Einfluss beispielsweise die Qualität der Leitung zu und vom Provider auf das Testergebnis hat, lässt sich nur bedingt abschätzen. »Für Cloud Testing ist die Fähigkeit, konsistente Ergebnisse zu erzielen, der kritische Faktor«, sagt CSC-Managerin Annette Gemeinder, »deshalb muss jeder an den Tests beteiligte Provider eine entsprechende Bandbreite und Verfügbarkeit sicherstellen können.«

Anderenfalls sind unter Umständen mehrere Testläufe zu unterschiedlichen Zeiten oder von verschiedenen Standorten nötig, um zu belastbaren Zahlen zu kommen, was den Aufwand deutlich erhöht.

Wichtig: Kontrolle

Selbstverständlich spielen auch beim Testen in der Cloud Datenschutz und -sicherheit eine große Rolle. Hier sind die anzulegenden Kriterien keine anderen als allgemein beim Transfer von Daten und Applikationen in die Cloud. Zertifizierungen nach Standards wie ISO 27001 können für einen Anbieter sprechen, ebenso wie Rechenzentrumsstandorte in Deutschland oder anderen Ländern der Europäischen Union. »Generell sind Testing-Dienstleister mit einer eigenen privaten Cloud gegenüber generischen Anbietern vorzuziehen, bei denen vieles anonym abläuft und oftmals nicht sehr transparent ist«, rät Georg Hansbauer von Testbirds.



Die Azure Dev-Test Labs von Microsoft lassen sich in die Entwicklungsumgebung Visual Studio Team Services integrieren (Bild 2)

Neben dem Datenschutz, der sich durch Anonymisierung oder Pseudonymisierung von Daten gewährleisten lässt, kommt dem Schutz des geistigen Eigentums beim Testen eine besondere Bedeutung zu. Nicht anders als in anderen Industrien ist der Wettbewerb in der Softwarebranche besonders an Neuentwicklungen und Prototypen der Konkurrenz interessiert.

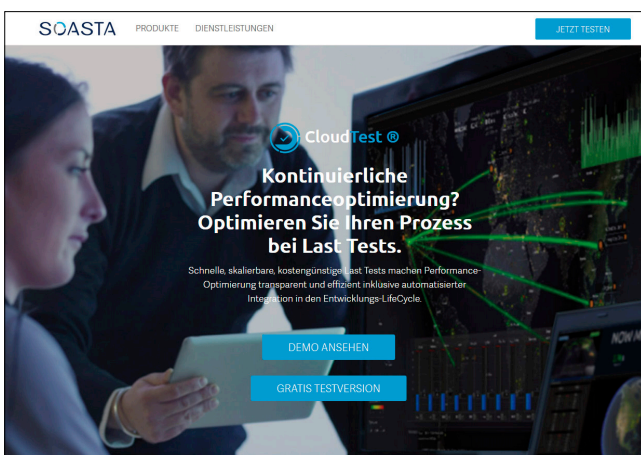
Um sensiblen Code schwerer lesbar zu machen, können Obfuskationsmethoden (von to obfuscate = vernebeln, verwirren) wie Code-Transformationen, Änderungen im Kontrollfluss, Redundanzen oder das Mischen von Funktionen zum Einsatz kommen. Eine Verschlüsselung ist ebenfalls denkbar.

Das alles erhöht natürlich wieder den Aufwand, der bei der Migration einer Testumgebung in die Cloud betrieben werden muss.

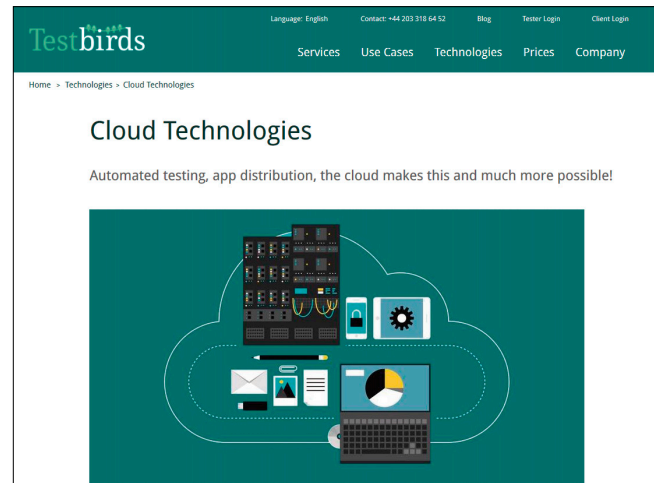
Für Matthias Zacher von IDC ist die Entscheidung, Testumgebungen in die Cloud outzusourcen, in erster Linie eine Vertrauensfrage: »Bin ich wirklich bereit, Cloud-Ressourcen zu nutzen? Das ist nicht nur in Testszenarien die entscheidende Frage.« Man könne sich zwar über Verträge und Service Level Agreements absichern, eine hundertprozentige Garantie gebe es jedoch nicht. »Das ist im eigenen Rechenzentrum aber auch nicht anders«, sagt Zacher. Ohnehin sei der Nutzer eines der größten Risiken, meint SOASTA-Manager Specht: »Ein Risiko also, welches man in jedweder Form von Mensch-Maschine-Kommunikation hat.«

Die Anbieter

Das Angebot an Test-Services ist umfangreich und wächst stetig. Neben großen Public-Cloud-Providern wie Amazon Web Services (AWS), Google oder Microsoft, IT-Vollsortimentern wie HPE und IBM, IT-Dienstleistern wie CSC oder der Capgemini-Tochter Sogeti und Test-Experten wie BlazeMeter oder SOASTA offerieren auch immer mehr kleine, spezialisierte Anbieter Tests als Service, zum Beispiel das deutsche Start-up Testcloud oder das Unternehmen Testfabrik, eine Ausgründung des Lehrstuhls für Softwaretechnik an der Universität des Saarlandes.



CloudTest von SOASTA unterstützt gleich 20 verschiedene Provider zur Lastgenerierung (Bild 3)



TestChameleon von Testbirds erlaubt das Testen verschiedener Betriebssystem- und Softwarekombinationen (Bild 4)

Die folgenden Beispiele erheben keinen Anspruch auf Vollständigkeit, sondern sie sollen vor allem die Breite der am Markt verfügbaren Dienste illustrieren.

Ein sehr umfangreiches Portfolio an Test-Services hat beispielsweise HPE (Bild 1). Hewlett Packard bietet schon seit mehr als zehn Jahren Testlösungen aus der Cloud an, die nach der Spaltung des Unternehmens im vergangenen Jahr nun an HPE übergegangen sind. Das Angebot reicht vom Projektmanagement für agile Softwareprojekte nach der SAFe-Methode (Scaled Agile Framework) über das Testmanagement, Last-, Performance- und Funktionstests bis hin zu Applikationssicherheitstests. Die Services lassen sich miteinander kombinieren und mit anderen Dienstleistungen wie Projekt- und Portfolio-Management, IT-Service-Management sowie Backup und Recovery verbinden. »Im Grunde genommen bieten wir alles, was ein Unternehmen braucht, um Anwendungen zu entwickeln, zu testen und zu betreiben«, sagt HPE-Manager Schütz.

Im Bereich Cloud Testing hat Microsoft verschiedene Optionen im Angebot, beispielsweise die Azure DevTest Labs, die sich in die Entwicklungsumgebung Visual Studio Team Services integrieren lassen (Bild 2). Der geschriebene Code wird automatisch kompiliert und auf einer Maschine im Azure DevTest Lab bereitgestellt, wo er entweder manuell oder automatisch getestet werden kann.

Ähnlich funktioniert die Xamarin Test Cloud für mobile Applikationen. »Diese tiefe und nahtlose Integration von Entwicklungs- und Test-Umgebung unterscheidet uns signifikant vom Wettbewerb«, sagt ALM-Architekt Karsten Kempe.

Mit seiner Lösung CloudTest unterstützt SOASTA gleich 20 verschiedene Provider zur Lastgenerierung. Die unterschiedlichen Clouds werden in ein eigenes Grid eingebunden, eine Art Cloud über der Cloud, die detaillierte Informationen in Echtzeit liefert. Der Anwender sieht im Dashboard unter anderem, welche Auswirkungen einzelne Elemente auf einer Webseite auf die Anwendung haben, und kann auf diese Weise »Störenfriede« isolieren. Während der Laufzeit des Tests lässt sich die Last verändern, pausieren oder für einzelne ►

Interview

»Theoretisch kann man jede Art von Softwaretest in der Cloud durchführen«

Dr. Alessio Gambi ist wissenschaftlicher Mitarbeiter am Lehrstuhl für Softwaretechnik an der Universität des Saarlandes und forscht dort über Testmöglichkeiten in der Cloud. Im Gespräch mit **web & mobile developer** erklärt er, worauf es beim Cloud Testing ankommt.

web & mobile developer: Herr Dr. Gambi, Sie beschäftigen sich schon seit einigen Jahren mit dem Thema Cloud Testing. Wie hat sich der Markt in Sachen Cloud Testing in den letzten Jahren entwickelt?

Alessio Gambi: Ich bin kein Marktforscher, aber aus meiner Perspektive als Wissenschaftler sehe ich ein großes Potenzial für Test-Services in der Cloud. Viele der allgemeinen Gründe, die Cloud-Services für Unternehmen attraktiv machen, gelten auch, wenn es um Softwaretests geht. Jedes Unternehmen, das sich mit Software-Entwicklung beschäftigt, ist ein potenzieller Kunde. Die Zielgruppe ist also riesig.

Man kann zur Beantwortung Ihrer Frage auch einfach die Zahl der Anbieter und der Dienste betrachten, die sich in letzter Zeit am Markt etabliert haben. Als ich meine Doktorarbeit über Cloud Testing geschrieben habe, gab es nur wenige Provider. Heute sind es sehr viel mehr und der Trend ist ungebrochen. Ich denke, das wird weiter zunehmen.

web & mobile developer: Warum sollte man überhaupt Softwaretests in die Cloud auslagern?

Gambi: Dafür gibt es drei gute Gründe. Als Erstes sind Kostenvorteile zu nennen. Man spart sich die fixen Aufwendungen für eine eigene Testumgebung und zahlt stattdessen nur für den Service, wenn man ihn braucht. Der zweite gute Grund ist die Skalierbarkeit, die Elastizität eines Cloud-Service. Gerade beim

Testfälle reduzieren. So kann der Tester Fehler korrigieren oder verschiedene Einstellungen ausprobieren, ohne dass er den Versuch abbrechen und neu starten müsste (Bild 3).

»Das reduziert die Anzahl an Test-Iterationen drastisch«, sagt SOASTA-Manager Specht. Lasttests sind auch im Produktivbetrieb möglich. Ein Real User Monitoring überwacht dabei ständig, ob echte Benutzer von den simulierten Zugriffen beeinträchtigt sind, und meldet das Feedback der Anwender in Echtzeit. Bei Problemen kann der Tester schnell reagieren und zum Beispiel die Last reduzieren. »Oft werden auf diese Weise Artefakte sichtbar, mit denen man nicht gerechnet hat«, sagt Specht, etwa Probleme in anderen Applikationen oder in Bereichen, die gar nicht im Fokus des Tests standen.



Dr. Alessio Gambi

Wissenschaftlicher Mitarbeiter
am Lehrstuhl für Software-
technik an der Universität des
Saarlandes
www.uni-saarland.de

Testen ändern sich die Anforderungen oft in kurzer Zeit sehr stark, etwa wenn das geplante Release-Datum näher rückt. Man müsste intern viele Ressourcen aufbauen, die man die meiste Zeit gar nicht nutzt, um solche Lastspitzen abzudecken. Der dritte große Vorteil liegt in der Plattformvielfalt, die man für Tests in der Cloud buchen kann. Über Device-

Farmen haben Sie beispielsweise Zugriff auf verschiedenste Modelle mobiler Endgeräte, auf diverse Plattformen und Betriebssystemversionen. Ähnlich ist es bei Cross-Browser-Kompatibilitätstests, wie sie etwa der deutsche Provider Testfabrik anbietet. Auch hier ist es für eine kleine Firma kaum möglich, eine Webseite oder Webapplikationen selbst auf allen möglichen Browsern, Betriebssystemen und Versionsständen zu testen.

web & mobile developer: Welche Nachteile handle ich mir ein, wenn ich Tests in die Cloud auslagere?

Gambi: Wie immer beim Outsourcing gibt man bei der Verlagerung von Tests in die Cloud einen Teil der Kontrolle über die firmeneigenen Prozesse ab. Das kann aus Sicherheits- oder Compliance-Gründen problematisch sein.

Die Integration von Cloud-Services in bestehende Testprozesse ist außerdem mit einem nicht zu unterschätzenden Aufwand verbunden. Die meisten Provider bieten dafür programmatische APIs als Schnittstellen. Der Anwender muss seine Testumgebung so umbauen, dass er die Cloud-

Das Testen verschiedenster Betriebssystem- und Softwarekombinationen ist die Spezialität der SaaS-Lösung Test-Chameleon von Testbirds. Über einen Webkonfigurator bestimmt der Kunde die Anzahl an virtuellen Maschinen mit den gewünschten Betriebssystemen und Browsern. Zusätzlich kann er weitere Softwarepakete wie Java-Versionen, aber auch Client-Software oder Virens Scanner auswählen (Bild 4). Manuelle Tests lassen sich über das HTML5-Frontend starten. Für die Automatisierung stehen neben einem generischen REST-API Schnittstellen zu Tools wie Selenium, Appium und Sikuli zur Verfügung. Jede Testumgebung wird separat aufgebaut und nach Testabschluss wieder zerstört. »So kann sie von niemand anderem außer dem Kunden selbst genutzt werden«, sagt Testbirds-Geschäftsführer Hansbauer.

Services über diese APIs ansprechen kann. Und schließlich haben Sie bei Tests in der Cloud nicht notwendigerweise Zugang zu allen Informationen, die Sie für die Interpretation der Testergebnisse benötigen. Der Provider stellt vielleicht einen Crawler oder einen Client zur Verfügung, aber zum Beispiel nicht den vollständigen Zugriff auf sämtliche Logfiles. Wenn die Entwickler aber gerade auf diese Information angewiesen sind, dann haben sie ein Problem, das womöglich erst klar wird, wenn man den Service bereits gebucht und integriert hat.

web & mobile developer: *Bei welchen Tests treten solche Schwierigkeiten denn am wenigsten auf?*

Gambi: Unkritisch sind sicher Last- und Stresstests, weil dabei weder Quellcode oder das Programm in die Cloud geladen werden müssen noch API-Anpassungen oder Ähnliches notwendig sind.

web & mobile developer: *Gibt es auch Testtypen, die prinzipiell nicht für eine Cloud-Verarbeitung geeignet sind?*

Gambi: Theoretisch kann man jede Art von Softwaretest in der Cloud ausführen. Die Frage ist natürlich, wie viel das Unternehmen über seine Applikationen und den Stand der Entwicklung offenbaren will. Das ist ja geistiges Eigentum, das unter Umständen geschäftskritisch ist. Wenn also sensible Informationen hochgeladen werden müssen, dann sollte man sich das gut überlegen oder gegebenenfalls Methoden der Code-Obfuskation verwenden.

web & mobile developer: *Bei personenbezogenen Daten spielt ja auch der Datenschutz eine Rolle ...*

Gambi: Ja, dem muss ich natürlich auch Rechnung tragen, etwa indem ich die Testdaten anonymisiere. Unter Umständen wird aber der Aufwand dadurch so groß, dass sich der Transfer gar nicht mehr lohnt. Auch wenn ich beispielsweise sehr große Datenbanken für meine Tests benötige, lohnt sich der Transfer in die Cloud womöglich nicht, zumal ich dort für die Datenhaltung entsprechend viel bezahlen müsste.

Fazit

Die Verlagerung von Tests in die Cloud bringt eine Vielzahl von Vorteilen. Einige davon wie Skalierbarkeit, Flexibilität und Verschiebung der Kosten von der Investitions- hin zur operativen Seite sind typische allgemeine Vorteile der Cloud-Nutzung.

Es gibt aber zusätzlich eine ganze Reihe spezifischer Argumente, die für die Cloud als Testplattform sprechen. Mobile Anwendungen beispielsweise lassen sich mit Hilfe von Device-Farmen auf einer großen Zahl von Plattformen, Betriebssystemversionen und Gerätetypen testen. Auch die Simulation weltweit verteilter Zugriffe und unterschiedlichster Lastszenarien ist ohne Cloud-Einsatz kaum sinnvoll möglich. Wirklich spannend wird das Testen in der Cloud aber erst,

web & mobile developer: *Nach welchen Kriterien sollte ein Unternehmen einen Cloud-Testing-Service auswählen?*

Gambi: Zunächst einmal muss klar sein, was genau ich von der Testplattform erwarte, welche Reportings oder Log-Dateien ich benötige, um das Testergebnis interpretieren zu können. Es gibt viele Angebote, die sehr ähnlich klingen, aber dann im Detail doch sehr unterschiedliche Aspekte der Softwarequalität untersuchen. Ein Kunde sollte außerdem unbedingt darauf achten, dass er mit dem Transfer seines Testprojekts in die Cloud nicht in eine Falle gerät, aus der er nicht wieder herauskommt. Als Google beispielsweise seine Projektverwaltungsplattform Google Code schloss, hatten alle Entwickler ein Problem, die sich darauf verlassen hatten.

web & mobile developer: *Der Preis spielt bei der Auswahl eines passenden Cloud-Testing-Services doch sicher auch eine Rolle, oder?*

Gambi: Ja, aber das ist ein schwieriges Thema, da sich die Angebote nur schwer vergleichen lassen und die Kosten von verschiedensten Parametern wie benötigter Rechenleistung, gebuchten Services, Kosten für Datenspeicherung und -transfer sowie den Vertragslaufzeiten abhängen können.

Da sich der Testaufwand vorab nur bedingt einschätzen lässt, kann man die Fragen nach eben diesen Parametern im Vorfeld nur grob beantworten. Ist er sehr viel höher als geplant, dann kommt der Service in der Cloud den Kunden unter Umständen sogar teurer, als wenn er eigene Systeme aufgebaut und verwendet hätte.

Immerhin erlauben Service-Provider wie Amazon Web Services eine grobe Kostenkalkulation. Man gibt den geplanten Aufwand an Maschinen, Speicher und Transferdaten in ein Webformular ein und erhält daraufhin eine Summe oder sogar verschiedene Service-Kombinationen und deren jeweilige Aufwände.

Generell kann man sicherlich sagen: Je genauer und je langfristiger man seinen Aufwand planen kann, desto kostengünstiger wird es.

wenn man es mit Konzepten wie der agilen Software-Entwicklung oder DevOps kombiniert und sämtliche Möglichkeiten der Testautomatisierung ausschöpft. ■



Dr. Thomas Hafen

ist seit mehr als 15 Jahren als Redakteur und Journalist tätig, unter anderem für die IT-Fachzeitschriften NetworkWorld Germany und ChannelPartner sowie die Fotoseiten Seen.by und Digitalkamera.de.

<http://thomas-hafen.de>

TWIG: TEMPLATE-SYSTEM FÜR PHP

Mächtiger dank Vorlagen

Mit Twig lässt sich die Trennung von Skriptcode und HTML unter PHP leicht realisieren.

Die Auftrennung in Skript- und Darstellungscode hat neben der besseren Übersichtlichkeit noch den Vorteil, dass Sie Webdesigner direkt an den Templates arbeiten lassen können, denn sie sehen – bis auf die Variablen-Platzhalter – nur den Teil des Projekts, der sie interessiert.

Man nun könnte meinen, dass mit PHP ein solches Template-System unnötig ist. Denn schließlich kann man doch auch mit PHP selbst ganz einfach Variablenersetzung nutzen. Doch das Problem mit PHP beginnt schon mit der Optik: Um einen Variableninhalt auszugeben, braucht man eine ausführliche Anweisung wie:

```
echo $myvar
```

In einem Twig-Template geht das um einiges kompakter, nämlich mit `{{ myvar }}`.

Wenn dann Manipulationen wie Escaping oder Zeichensatz-Änderungen notwendig werden, ist der Unterschied noch eklatanter. Hier die für HTML aufbereitete Ausgabe einmal mit PHP:

```
echo htmlspecialchars($myvar, ENT_QUOTES, 'UTF-8')
```

Und hier das Gegenstück in Twig:

```
{{ myvar|escape }}
```

Und dabei kann sogar der Teil `lescape` in den meisten Fällen weggelassen werden.

Twig-Templates sind einfache Textdateien und haben nicht zwingend HTML als Zielformat. Sie können im Prinzip damit alle Dateiarten erzeugen, die auf Text aufgebaut sind. Es gibt auch keine vorgeschriebene Extension. Wir verwenden in den Beispielen aber die Endung `.twig`, weil so die meisten IDEs erkennen, dass es sich um ein Twig-Template handelt, und daraufhin Funktionen wie Syntax-Highlighting und automatische Ergänzung von Schlüsselwörtern aktivieren.

So funktionieren Twig-Templates

Die einfache Variablensubstitution haben Sie ja schon kennengelernt. Dabei zeigt das Auftauchen einer doppelten geschweiften Klammer dem System an, dass es hier etwas ersetzen soll. Eine geschweifte Klammer gefolgt von einem Prozentzeichen leitet eine Kontrollstruktur ein, etwa eine Bedingung oder eine Schleife:

```
<h3>Ihre Einkäufe</h3>
{% for produkt in warenkorb %}
```



Von der Dokumentation für Twig gibt es spezielle Varianten für Webdesigner und Programmierer (Bild 1)

```
<p>{{produkt}}</P>
{% endfor %}
```

Die letzte Art der Klammerung von Twig dient zum Einfügen von Kommentaren.

Damit Ihre Programme überhaupt mit einem Twig-Template umgehen können, müssen Sie zuerst das Grundsystem installieren – etwa durch Composer mit `composer require twig/twig`. Danach genügt es, den von Composer zur Verfügung gestellten Autoloader einzubauen, um Twig nutzen zu können:

```
include 'vendor/autoload.php';
```

Falls Sie Composer nicht einsetzen möchten, holen Sie sich das Archiv von <http://twig.sensiolabs.org>, entpacken es und verwenden den eigenen Autoloader von Twig (Bild 1):

```
include '/ihr/pfad/Twig/Autoloader.php';
Twig_Autoloader::register();
```

Die Initialisierung von Twig erfolgt in zwei Schritten: Zuerst rufen Sie ein Objekt von Typ `Twig_Loader_Filesystem` ins Leben. Das erhält als Parameter das Verzeichnis, in dem es die Templates suchen soll. Sie können anstelle eines Namens auch ein Array von Verzeichnisnamen verwenden, dann durchsucht Twig nacheinander die genannten Verzeichnisse

nach einem Template, bis es fündig wird. Im nächsten Schritt dann wird dem Twig-Objekt selbst Leben eingehaucht:

```
$twig = new Twig_Environment($loader);
```

Diese Methode sieht ein wenig umständlich aus. Man könnte doch schließlich einfach dem Twig-Objekt den Pfad mitgeben. Der Vorteil der Zwischenschaltung eines Loader-Objekts ist, dass Sie damit frei mit der Wahl des Speicherorts für Ihre Templates sind. So gibt es beispielsweise einen Loader für Templates in einer Datenbank. Abkürzen können Sie das Ganze immerhin dadurch, dass Sie auf die zwischengeschaltete Variable `$twigLoader` verzichten:

```
$twig = new Twig_Environment
(new Twig_Loader_Filesystem(__DIR__ . "/templates"));
```

Um ein Template mit Leben zu füllen, verwenden Sie eine Zeile wie

```
$twig->display('bsp.twig', ['user' => $myuser];
```

Der erste Parameter für die Methode `display()` gibt dabei den Namen des Template-Files an. Den sucht Twig dann im Verzeichnis, das beim Initialisieren angegeben wurde. Der zwei-

Wenn Sie also zum Beispiel `user.vorname` schreiben, prüft die interne Logik zuerst, ob es sich um bei `user` um ein Array handelt. Falls ja, dann wird das Element mit dem Schlüssel `vorname` darin verwendet. Ist `user` dagegen ein Objekt, dann versucht Twig die Instanzvariable `vorname` anzusprechen. Gibt es auch die nicht, dann wird geprüft, ob es im Objekt eine Methode `vorname()` gibt. Falls ja, wird sie aufgerufen und ihr Rückgabewert ausgegeben.

Es gibt noch ein paar weitere Versuche, an das Element `vorname` zu gelangen. Der Screenshot aus der Twig-Doku zeigt die komplette Liste, die Twig bei der Interpretation von attribuierten Variablen mit abarbeitet (Bild 2).

Filter vereinfachen die Nutzung

Über Filter manipulieren Sie Variableninhalte in kompakter Schreibweise. Filter werden mittels eines Pipe-Symbols an einen Variablennamen angehängt. So gibt es zum Beispiel einen Filter `trim`, der wie die gleichnamige PHP-Funktion Leerzeichen am Anfang und Ende einer Zeichenkette abschneidet. Filter können auch verkettet werden, indem man einfach noch ein weiteres Pipe-Zeichen und einen Filternamen anhängt, um den Namen zum Beispiel zusätzlich noch in Großbuchstaben umzuwandeln.

Sie können mit den Filtern aber noch mehr machen als nur diese einfachen Konvertierungen. So gibt es einen Filter `join`,

der die Elemente eines Arrays mit einem Separator verknüpft. Besteht zum Beispiel die Variable `tags` aus einem Array von Strings, dann geben Sie mit folgendem Konstrukt den Inhalt des Arrays durch Kommas separiert aus:

```
{{ tags|join(, ,) }}
```

Bei solchen Filtern mit Parameter wird also dieselbe Syntax mit runden Klammern verwendet wie bei Funktionsaufrufen in PHP.

Ganz praktisch für Fälle wie Ajax-Backends ist der Filter `json_encode()`. Wenn Sie zum Beispiel in Ihrem Template schreiben:

```
{{ user|json_encode() }}
```

dann macht Twig daraus folgende Ausgabe:

```
{"name":"M\u00fcller","vorname":"Peter","ort":"Berlin"}
```

Bei der Ausgabe von Variableninhalten gibt es immer das Problem, dass das Zielsystem Teile des Inhalts als Befehl interpretieren könnte. Ein Beispiel: Als Firmenname soll *Bang & Olufsen* ausgegeben werden, der Browser versucht das `&` aber als Beginn einer HTML-Entity zu verarbeiten. Darum setzt man Escaping ein, das die kritischen Zeichen oder Zeichenfolgen so umsetzt, dass die Ausgabe korrekt ist.

Standardmäßig ist Twig in der aktuellen Version so konfiguriert, dass es automatisch Escaping für HTML durchführt. Wenn Sie also in einer String-Variablen ein Zeichen wie `<` ►

Implementation

For convenience's sake `foo.bar` does the following things on the PHP layer:

- check if `foo` is an array and `bar` a valid element;
 - if not, and if `foo` is an object, check that `bar` is a valid property;
 - if not, and if `foo` is an object, check that `bar` is a valid method (even if `bar` is the constructor - use `__construct()` instead);
 - if not, and if `foo` is an object, check that `getBar` is a valid method;
 - if not, and if `foo` is an object, check that `isBar` is a valid method;
 - if not, return a null value.
- `foo['bar']` on the other hand only works with PHP arrays:
- check if `foo` is an array and `bar` a valid element;
 - if not, return a null value.

Twig probiert beim Auftauchen des Variablen-Attributs eine Menge sinnvoller Fälle durch, um an die richtigen Daten zu gelangen (Bild 2)

te Parameter ist das Array, das die zu übergebenden Variablen festlegt. Innerhalb des Template-Kontexts gibt es also eine Variable `user`, die der Variable `$myuser` im aufrufenden Skript entspricht. Mit der Methode `render()` gibt es eine Variante von `display()`, die genauso aufgerufen wird, aber das Ergebnis nicht direkt anzeigt, sondern zurückliefert, sodass Sie es in etwa einer Variablen zwischenspeichern oder sonst weiterverwenden können.

Zugriff auf komplexere Variablen

Twig kann nicht nur mit einfachen Variablen umgehen, sondern verarbeitet genauso Arrays oder Objekte. Twig ist so konzipiert, dass Sie mit diesem Konzept alle sinnvollen Zugriffe auf Arrays und Objekte erledigen können.

haben, dann macht es daraus automatisch *<*. Das ist praktisch, weil Sie sich so nicht um die Aufbereitung kümmern müssen. Twig kennt aber noch weitere Escape-Strategien, etwa für Javascript, CSS oder URLs. Möchten Sie nur eine einzelne Variable damit bearbeiten, dann nutzen Sie den Filter *escape*. Er erhält als Parameter die gewünschte Strategie:

```
{{ user.name|escape('js') }}
```

Wenn Sie in Ihrem Template einen ganzen Block mit JavaScript haben, ist natürlich trotzdem nicht praktikabel, wenn Sie bei jeder Variablen diesen Filter einfügen müssen. Dazu können Sie das Escaping grundsätzlich auf eine andere Methode umstellen:

```
{% autoescape 'js' %}
```

Das Kommando *endautoescape* stellt dann wieder den vorigen Zustand her.

Manchmal ist es auch notwendig, dass keinerlei Aufbereitung der Variablenwerte passiert, etwa bei der Generierung

von XML. Dieses Verhalten können Sie über den Filter *raw* bei einer Variablenausgabe bewirken:

```
{{ mystring|raw }}
```

Oder Sie stellen das Escaping für mehrere Zeilen aus. Dazu setzen Sie folgende Direktive ein:

```
{% autoescape false %}
```

Auch hier gilt die Einstellung bis zum Ende des Templates oder bis Twig auf die folgende Zeile trifft:

```
{% endautoescape %}
```

Wenn Sie mit den vorgegebenen Escapern nicht zurechtkommen, können Sie sich auch einen eigenen basteln. Dazu schreiben Sie sich eine Funktion, die die notwendigen Umwandlungen durchführt, und registrieren sie nach dem Instanzieren des Twig-Objekts:

```
$twig->getExtension('core')->setEscaper(
    'myesc', 'my_escaper');
```

Wenn Sie dann innerhalb eines Templates eine Variable so verwenden:

```
{{ mystring|e('myesc') }}
```

dann ruft Twig Ihre Funktion *my_escaper()* mit dem Variablenwert als einen von drei Parametern auf und setzt den zurückgelieferten Wert in das gerenderte Ergebnis ein.

Die anderen beiden Parameter, die Ihrer Escape-Funktion mitgegeben werden, sind das Twig-Objekt und der Zeichensatz.

Kontrollstrukturen für Templates

Sie brauchen in der Regel auch in einer Template-Datei Bedingungen, Schleifen und andere Elemente zur Steuerung. Diese sind bei Twig an ihre Pendanten aus PHP angelehnt. So gibt es beispielsweise das Konstrukt *{% if ... %}*:

```
{% if user.premiumkunde %}
<p>Sie erhalten Ihr Eilpaket schon morgen!</p>
{% else %}
<p>Sie erhalten Ihr Paket in drei Tagen.</p>
{% endif %}
```

Ebenfalls nützlich sind *for*-Schleifen, die Sie auf verschiedene Weisen einsetzen können.

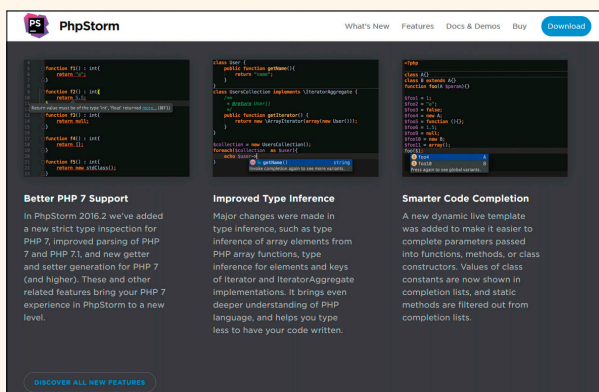
Ein ziemlich mächtiges Werkzeug von Twig ist die Vererbung. Diese funktioniert in der Weise, dass Sie in einem Template einen Block definieren, der von Ihnen einen Namen enthält:

```
{# Dies ist die Datei layout.twig#}
{% block navi %}
```

IDE-Integration

Einige PHP-Entwicklungsumgebungen können nativ mit Twig umgehen, wie etwa PhpStorm, für viele andere gibt es entsprechende Plug-ins:

- Textmate – via Twig-Bundle
- Vim – via Jinja syntax-Plug-in oder das vim-twig-Plug-in
- NetBeans – via Twig Syntax Plug-in (bis 7.1, nativ ab 7.2)
- PhpStorm – nativ ab 2.1
- Eclipse – via Twig-Plug-in
- Sublime Text – via Twig-Bundle
- GtkSourceView – via Twig-Language-Definition
- Coda und SubEthaEdit – via Twig-Syntax-Mode
- Coda 2 – via Twig-Syntax-Mode
- Komodo und Komodo Edit – via Twig Highlight/Syntax-Check-Mode
- Notepad++ – via Notepad++ Twig Highlighter
- Emacs – via web-mode.el
- Atom – via PHP-twig for atom



Die IDE PhpStorm kann nativ mit Twig umgehen

Wichtige Startoptionen für Twig

Der zweite, optionale Parameter beim Anlegen des `Twig_Environment`-Objekts ist ein Array mit Optionen, mit denen Sie die Funktionsweise der Bibliothek steuern können. Hier eine Liste der wichtigsten Optionen des Systems:

- **strict_variables:** Trifft Twig in einem Template auf eine nicht definierte Variable, übergeht es normalerweise das Problem und ersetzt sie einfach mit `null`. Es wird also dort gar nichts ausgegeben. Das trifft genauso auf nicht existierende Variablen-Attribute zu. Setzen Sie `strict_variables` auf `true`, wirft Twig stattdessen eine Exception, um Ihnen das Problem aufzuzeigen.
- **charset:** Standardmäßig geht Twig davon aus, dass die Daten, die Sie damit verarbeiten, in UTF-8 codiert sind. Mit diesem Schalter können Sie das zum Beispiel auf ISO-8859-15 umstellen. Wollen Sie dagegen lediglich für eine einzelne Variable eine bestimmte Umcodierung erreichen, verwenden Sie dazu den Filter `convert_encoding`. Er benötigt als ersten Parameter die gewünschte Zielcodierung und als zweiten die Quellcodierung.
- **autoescape:** Standardmäßig ist bei Twig das Autoescaping aktiviert und HTML als Kontext gewählt. Mit der Einstellung `autoescape` können Sie dies generell umstellen und zum Beispiel CSS

als Standardkontext wählen. Möchten Sie diese Funktion ganz deaktivieren – etwa zur Ausgabe von XML –, dann setzen Sie den Wert auf `false`.

- **cache:** Wenn Sie hier den Pfad eines Verzeichnisses setzen, das der Webserver beschreiben darf, nutzt Twig Caching. Das bedeutet, dass Ihre Templates dort in kompilierter Form zwischengespeichert werden, was die Ausführungsgeschwindigkeit bei allen Folgezugriffen erhöht. Laut dem Hersteller soll der Unterschied bei aktiviertem Opcode-Caching, also etwa mit APC, noch deutlicher ausfallen.
- **debug:** Für die Fehlersuche in Templates sind zwei Änderungen vorzunehmen. Einerseits setzen Sie `debug` auf `true`. Zusätzlich erfordert Twig aber noch, dass Sie die Erweiterung `Twig_Extension_Debug` aktivieren, damit die Ausgabe von Informationen über Variablen aus einem Template heraus möglich ist. Nun können Sie in einem Template die Funktion `dump()` verwenden, um den Inhalt einer Variablen im Stil von `var_dump()` ausgeben zu lassen. Für bessere Lesbarkeit können Sie das in ein `<pre>`-Tag einschließen. Verwenden Sie `dump()` ohne Parameter, gibt Ihnen das System alle dem Template bekannten Variablen aus.

```
<li><a href="/">Home</a></li>
<li><a href="/search">Suche</a></li>
{% endblock %}
```

Ein anderes Template kann nun diese Datei verwenden, dabei aber den Block gezielt überschreiben:

```
{# Dies ist die Datei admin.twig#}
{% extends "layout.twig" %}
{% block navi %}
<li><a href="/">Home</a></li>
<li><a href="/pma">DB-Verwaltung</a></li>
{% endblock %}
```

Wenn Sie also das Template `admin.twig` rendern lassen, zeigt es im Prinzip dieselbe Information an wie `layout.twig`, tauscht aber dessen Block `navi` gegen seine eigene Version aus.

Möchten Sie innerhalb eines vererbten Templates Zugriff auf den Inhalt des Vorgängers haben, dann verwenden Sie die Funktion `parent()`. Folgende Version von `admin.twig` würde beispielsweise die originale Navigation anzeigen, aber daran noch zwei Menüpunkte anhängen:

```
{% extends "layout.twig" %}
{% block navi %}
{{ parent() }}
<li><a href="/pma">DB-Verwaltung</a></li>
<li><a href="/stats">Statistik</a></li>
{% endblock %}
```

Das eröffnet Ihnen vielfältige Möglichkeiten für die Modularisierung Ihres Weblayouts.

Sandboxing für mehr Sicherheit

Setzen Sie möglicherweise Templates aus unsicheren Quellen ein, hilft Ihnen die Sandbox-Extension dabei, dass Ihre Anwendung dadurch nicht gefährdet wird. Dazu definieren Sie ein Policy-Objekt, das festlegt, welche Tags, Filter, Objekt-Methoden, Objekt-Properties und Funktionen erlaubt sein sollen:

```
$policy = new Twig_Sandbox_SecurityPolicy($tags,
    $filters, $methods, $properties, $functions);
```

Dieses Regelwerk wird dann als Parameter für die Initialisierung eines Sandbox-Objekts verwendet und das als Extension hinzugefügt:

```
$sandbox = new Twig_Extension_Sandbox($policy,true);
$twig->addExtension($sandbox);
```

Sobald danach eines der geladenen Templates einen nicht erlaubten Zugriff versucht, wirft Twig die Exception `Twig_Sandbox_SecurityError`. ■



Markus Schraudolph

ist Journalist und Programmierer. Er schreibt seit 16 Jahren Bücher und Artikel für Fachzeitschriften. Seine Schwerpunktgebiete als Programmierer sind die Webprogrammierung und Datenbanken.

OFFENE STANDARDS VEREINFACHEN DIE INTEGRATION EXTERNER DATEN

Integration

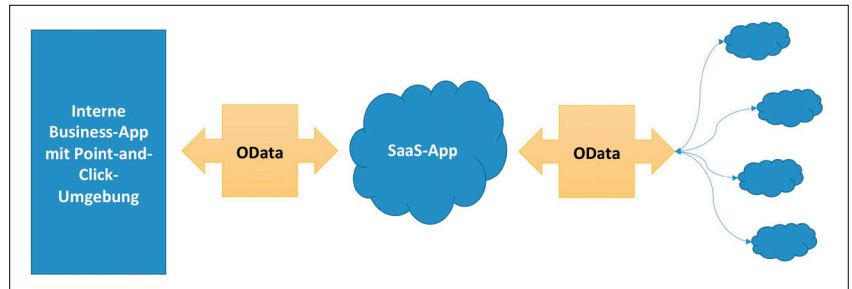
Offene Standards wie OData erleichtern die Einbindung externer Datenquellen.

Die Trendthemen Cloud und Mobile haben dafür gesorgt, dass in immer mehr Unternehmen Fachbereiche wie Marketing, Vertrieb und Support die unterschiedlichsten Datenformate einsetzen. Die Fachanwender nutzen SaaS-Anwendungen wie das Cloud-basierte CRM-System Salesforce, die Marketing-Automation-Software Marketo oder die Servicemanagement-Lösung ServiceMax. Das Problem beim Einsatz solcher SaaS-Anwendungen ist die Integration in bereits vorhandene Anwendungen, aber auch das Zusammenspiel verschiedener Cloud-Applikationen untereinander.

Anforderungen an eine Strategie zur Integration externer Daten

Eine erfolgreiche Integration externer Daten ermöglicht den Zugriff auf heterogene Datenquellen, ohne dass die Informationen lokal kopiert werden müssen. Entscheidend ist, dass der Datenaustausch in beiden Richtungen möglich ist und somit alle beteiligten Anwendungen auf denselben Datenbestand zugreifen.

Die starke Verbreitung der Cloud hat drei Randbedingungen hervorgebracht, die in einer Datenintegrationsstrategie berücksichtigt werden müssen: Citizen Development, Verknüpfung hybrider Daten und Verbindungen über das Web:



Quelle: Progress

SaaS-Anwendungen ermöglichen mit OData Abfragen des eigenen Datenbestands und binden zugleich andere Cloud-Services ein (Bild 1)

- Citizen Development bezeichnet die Entwicklung von Anwendungen durch Nutzer aus Fachbereichen anstatt aus der IT. Die Anwender setzen dafür allerdings nicht die herkömmlichen Entwicklungsumgebungen ein, sondern beispielsweise spezielle Point-and-Click-Plattformen, die von Entwicklern aus der IT eigens dafür vorbereitet werden.
- Die Verknüpfung hybrider Daten wird immer wichtiger, da sich ein wachsender Teil der vom Unternehmen genutzten Daten jenseits der Corporate Firewall befindet. Mit jeder SaaS-Anwendung steigt die Vielfalt und Heterogenität der Datenstrukturen, die in On-Premise-Anwendungen integriert werden müssen.
- Die Verbindung zu SaaS-Datenquellen läuft über das Webprotokoll HTTP. Dafür bieten die Anwendungen ein API (Application Programming Interface), das den automatisierten Anwendungszugriff auf die Webapplikation erlaubt.

OData – Open Data Protocol

OData ist ein von Microsoft definiertes und veröffentlichtes Protokoll zur Abfrage und Änderung von Daten über das Web. OData ist ein einheitliches Modell für den Datenaustausch in der Client-Server-Kommunikation. Es nutzt die Standards HTTP, ATOM und JSON für die Datenübertragung.

Microsoft hatte bereits im Jahr 2007 damit begonnen, eine allgemeine Lösung für datenbasierte Webservices zu entwickeln. Hieraus entstand nach kurzer Zeit die Initiative eines offenen, plattformneutralen Protokolls, das von jedem Entwickler in seine Services und Webapplikationen eingebunden werden kann.

Seit März 2014 ist das Open Data Protocol in Version 4.0 und das OData JSON Format in Version 4.0 offiziell als Standard der OASIS (Organization for the Advancement of Structured Information Standards) anerkannt. Inzwischen gibt es Entwickler-Kits für .NET, PHP, SAP und Java.

Aus diesen Aspekten lassen sich drei wichtige Anforderungen an eine Strategie für die Integration externer Daten ableiten. Erstens sollten alle Tools für die Datenintegration mit einer intuitiv verständlichen Point-and-Click-Bedienoberfläche ausgestattet sein. Zweitens sollte das Integrationswerkzeug möglichst viele gängige Datenquellen berücksichtigen, etwa klassische relationale Datenbanken, gängige Datenformate wie etwa EDI oder CDV, moderne Big-Data-Systeme wie Hadoop, SAP HANA oder MongoDB und schließlich die Datenformate der SaaS-Marktführer. Die dritte und letzte Anforderung ergibt sich aus dem Einsatz von APIs, die neben dem Datenformat ebenfalls in eine Lösung integriert werden.

OData, ein Standard für den Datenaustausch

Einen einfachen Weg zu diesem Ziel gibt es allerdings nicht. API-Zugriffe auf externe Datenquellen stehen im Konflikt mit der Netzwerksicherheit und sind für Anwender aus den Fachbereichen normalerweise verboten. Es fehlt also noch ein ent-

scheidendes Element: ein leicht zu nutzender Standard für den Datenaustausch. Er sollte ein interoperables API bieten, einen bidirektionalen Datenzugriff erlauben und einfach in Anwendungen, aber auch in Point-and-Click-Interfaces integriert werden können. Ein solcher Standard existiert: Es ist das Open Data Protocol oder kurz OData (Bild 1).

OData nutzt die offenen REST-APIs, die in vielen SaaS-Anwendungen integriert sind. Die Abkürzung REST steht für

Zugriff darauf möglich ist. Damit müssen sie keine Konnektoren für gängige On-Premise-Applikationen bereitstellen. Gleichzeitig sind sie in der Lage, mit geringem Aufwand eigene Routinen für die Datenintegration weiterer Anbieter zu erstellen.

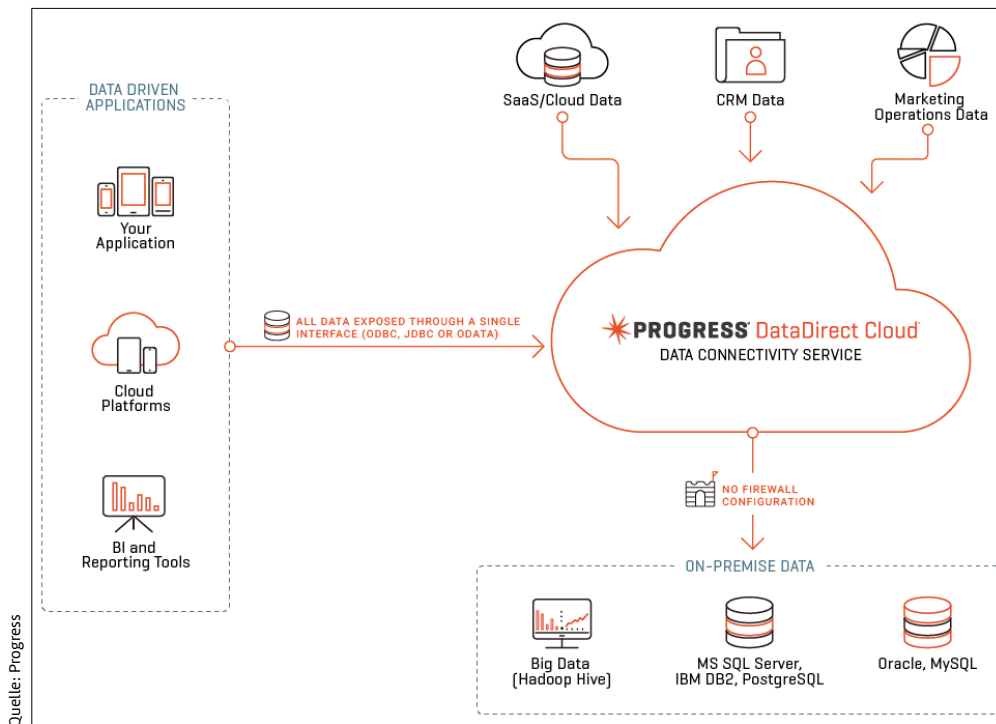
- Auch für die Fachanwender hat ein Standard wie OData Vorteile. So können Unternehmen, die mehrere SaaS-Applikationen einsetzen, sehr leicht selbst Schnittstellen für die Integration der Daten entwickeln. Zudem werden keine Konnektoren von Drittherstellern mehr benötigt.

Eine technisch effiziente Implementierung der Datenkonnektivität mit OData ist vor allem dann entscheidend, wenn die Zugriffe von Anwendern aus den Fachbereichen konfiguriert werden. Dadurch kann die IT-Organisation deutlich entlastet werden und die Auftragsprogrammierung von abteilungs- oder projektspezifischen Abfragen entfällt.

Die beiden bekannten SaaS-Anbieter Salesforce und Oracle unterstützen OData seit Kurzem in ihren Lösungen. Sie erweitern damit die Gruppe der sogenannten Producer-Services, die eine OData-Schnittstelle anbieten. Da-

zu gehören neben verschiedenen Microsoft-Produkten wie SharePoint, Azure und Dynamics CRM sowie NAV auch das SAP NetWeaver Gateway, IBM DB2 und Informix sowie Progress DataDirect (Bild 2).

Hier zeigt sich deutlich die Leistungsfähigkeit des OData-Standards. Eine Connectivity-Suite wie DataDirect kann damit auf alle gängigen Backoffice-Datenquellen zugreifen – unabhängig von Art und Ort der Quelle. Die standardbasierte Strategie zur Integration externer Daten verschafft einem Unternehmen die notwendige Flexibilität bei der Datenverarbeitung. ■



Mit OData kann Progress DataDirect Cloud auf SaaS-Apps, Big Data und Social Data zugreifen (Bild 2)

Representational State Transfer und bezeichnet ein modernes Programmierparadigma für verteilte Systeme, das Struktur und Verhalten des Web nachbildet. REST-APIs ermöglichen Entwicklern, auf die Daten und Funktionen der Webservices zuzugreifen und sie in eigene Anwendungen zu integrieren.

Die APIs richten sich nicht nur an die Anwender von SaaS-Anbietern. Die Dienste nutzen ihre APIs auch gegenseitig, sodass sie sich zu Software-Ökosystemen vernetzt haben, bei denen verschiedene Lösungen miteinander verbunden werden können. Allerdings ist nicht jede theoretisch mögliche Datenintegration auch tatsächlich verfügbar, sondern meist nur einzelne Dienste.

Umfassende Lösung

OData ist dagegen eine umfassende Lösung, die API-Zugriffe auf externe Daten aus zahlreichen Quellen für Entwickler vereinheitlicht und vereinfacht. Der Standard bildet damit ein weit übergreifendes Ökosystem aus SaaS-Datenquellen. Er kann von zwei unterschiedlichen Anwendergruppen genutzt werden: SaaS-Anbietern und Fachanwendern:

- Die Anbieter von SaaS-Anwendungen können ihre Datenzugriffsschicht damit so entwickeln, dass ein einheitlicher



Sumit Sarkar

ist Chief Data Evangelist bei Progress, einem Anbieter von Applikations-Infrastruktur-Software für die Entwicklung, die Integration und das Management von Geschäftsanwendungen.

www.progress.de

GRAFIK FÜR ENTWICKLER

Grafik aus der Wolke

Cloud Computing und Online-Editoren – wie Bildbearbeitung über das Netz funktioniert.

Hier noch eine kleine Farbkorrektur, dort schnell die lästige Stromleitung über dem Bildmotiv weggestempelt – was im Büro mit Adobe Photoshop CC schnell gemacht ist, ist auf dem weniger gut bestückten Laptop nicht immer auf Anhieb möglich.

Auf den folgenden Seiten stellen wir verschiedene Online-Editoren vor, mit denen Bilder zumindest für digitale Anwendungen zufriedenstellend bearbeitet werden können.

Für bestimmte Arbeiten, etwa für eine anspruchsvolle Bildbearbeitung, sind in der Regel die notwendigen Programme wie Adobe Photoshop und Co. fest auf dem Computer installiert.

Um umfangreiche Arbeiten grafischer Natur auch im Home Office zu erledigen, müsste die entsprechende Software auch auf dem eigenen Rechner zur Verfügung stehen. Zudem sollten alle Daten bereitstehen, was heute dank Cloud Computing kein Problem mehr darstellt.

Cloud-Modelle

Cloud Computing stellt IT-Ressourcen über das Netz zur Verfügung, beispielsweise Software oder Speicherplatz. Dabei ist der Umfang der angebotenen Leistungen sehr unterschiedlich – und damit zusammenhängend unterscheiden sich auch die Gebühren der einzelnen Cloud-Anbieter für ihre Dienste (Bild 1).



Sind die Daten in der Cloud gespeichert, lassen sie sich flexibel von jedem Ort aus abrufen und bearbeiten (Bild 1)

Nach einer Definition des NIST (National Institute of Standards and Technology) beinhaltet Cloud Computing drei unterschiedliche Service-Modelle:

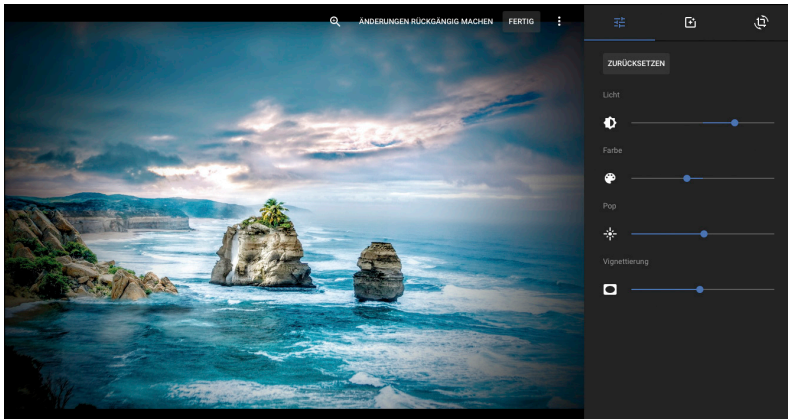
Das Modell IaaS (Infrastructure as a Service) stellt seinen Benutzern einen Verbund von Rechnern, einen sogenannten Computer-Cluster, bereit. Die Daten können hier mit der für das Projekt entsprechend notwendigen Rechnerleistung bearbeitet werden, selbst wenn an einer Stelle im Computer-cluster ein Fehler auftritt. Verantwortlich für das Betriebssystem und die Software ist der Anwender.

Eine Entwicklerumgebung mit allen notwendigen Werkzeugen und Diensten liefert das Modell PaaS (Platform as a Service). Dabei sollten alle Ressourcen für den Nutzer flexibel und skalierbar sein. Neben der Entwicklung der Software kann dieses Cloud-Modell auch die Möglichkeit bieten, das fertige Programm über einen Service-Provider bereitzustellen. Das SaaS-Modell (Software as a Service) offeriert dann eine oder mehrere Softwarelösungen über das Netz. Solche Programme werden von dem IT-Anbieter verwaltet und vom Anwender über einen Browser gestartet. Eine Installation auf dem eigenen Rechner ist somit nicht notwendig.

Die einzelnen Lösungen, die über das SaaS-Modell bereitstehen, siedeln sich hauptsächlich im kaufmännischen Bereich an. Einer der ersten Anbieter von Cloud-Computing-Software, die heute noch auf dem Markt positioniert sind, ist Salesforce.com: Zur Verfügung stehen sieben Cloud-Lösungen für Vertrieb, Support, Marketing, Kundenservice und Analyse. Verschiedene Lizenzen berücksichtigen Unternehmen jeder Größe (Bild 2).



Vorreiter von Cloud Computing war Salesforce.com mit seinen Lösungen zur Verwaltung für Unternehmen (Bild 2)



Die Bildverwaltungslösung Google Foto bietet nur wenige Filter zur Bearbeitung von Fotos (Bild 3)

Online-Lösungen von Google

Zwar wird Google hauptsächlich mit der gleichnamigen Suchmaschine, dem Browser Chrome oder dem Online-Kartendienst Google Maps in Verbindung gebracht, das US-amerikanische Unternehmen stellt jedoch auch verschiedene Dienste und Softwarelösungen online zur Verfügung. So bietet Google seinen eingeloggtten Nutzern mit Gmail ein eigenes E-Mail-Programm, als Ergänzung für die Organisation gibt es dann noch den Kalender und die Kontakte. Auch alle anderen Lösungen, wie sie beispielsweise das Microsoft-Office-Paket bietet, deckt Google als Online-Version ab und macht sie so über einen beliebigen Browser von überall aus zugänglich. Hierzu gehören Google Docs, Google Tabellen und Google Präsentationen genauso wie die ebenfalls kostenlose Lösung Google Sites zum Erstellen von Webseiten. Ergänzt werden diese Softwarelösungen durch den Online-Speicher Google Drive.

Weiter ermöglicht Google mit Hangouts die Kommunikation über Computer, Tablet oder Smartphone per Textnachricht und Video. Seit diesem Jahr ist auch der Google-Messenger Spaces verfügbar, ein soziales Netzwerk, das als eine Art Erweiterung von Hangouts gesehen werden kann.

Was viele nicht wissen: Auch das Unternehmen YouTube LLC ist ein Tochterunternehmen von Google. Doch wo bleiben nun die Mittel und Werkzeuge zur Bearbeitung von Bildern und Grafiken? Bis vor Kurzem hat Google diesen Bereich durch die Online-Bildverwaltungs- und Bearbeitungslösung Picasa abgedeckt, die schon länger nicht mehr zur Verfügung steht. Nachfolger ist das Tool Google Fotos, das jedoch eher als Cloud-Speicher und Verwaltung für Bilder und Videos gesehen werden kann. Die allernotwendigsten Filter liefert ein Editor dennoch: So lassen sich Licht, Farbe und Kontrast einstellen, auch eine Vignettierung ist möglich. Weiter gibt es verschiedene, in der Stärke einstellbare Farbfilter. Auch schiefe Bilder rückt die Funktion *Zuschneiden und Drehen* zurecht (Bild 3).

Adobe im Netz

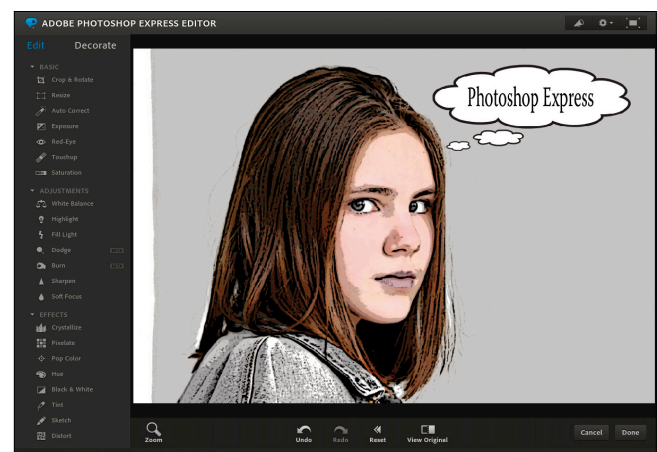
Mit der Creative Cloud möchte Adobe allen Vertretern der unterschiedlichsten Berufe im kreativen Bereich die Arbeit

erleichtern. So erlaubt es die Creative Cloud, Daten in der firmeneigenen Wolke abzulegen, um diese dann mit Kollegen zu teilen und sie von einem beliebigen Ort aus weiter zu bearbeiten.

Über die Creative Cloud lassen sich beispielsweise Schriften, Stile, Farben oder ganze Grafiken mit allen teilen, die über ein Creative-Cloud-Konto verfügen. Zudem ist der Zugriff auf die verschiedenen, vom Nutzer angelegten Online-Bibliotheken von fast jedem Programm aus möglich, was die Arbeit mit mehreren Tools ungemein erleichtert. Weiter steht über Adobe Stock professionelles Bild- und Videomaterial bereit.

Die einzelnen professionellen Lösungen wie Photoshop, Illustrator oder InDesign sind zwar bereits seit der ersten Version der Cloud nur noch als Abo-Modell online erhältlich, sie müssen jedoch, wie die Vorversionen der Adobe Creative Suite, vor dem Gebrauch geladen und installiert werden. Zwar kann so auch offline gearbeitet werden, aber – ob online oder offline – jedes einzelne Programm muss sich zunächst auf dem Computer befinden. Home-Office ist also nach wie vor nur bei entsprechendem Equipment möglich.

Um dennoch schnell das ein oder andere Bild am heimischen Rechner ohne Photoshop CC bearbeiten zu können, stellt Adobe einen Online-Editor über Photoshop.com bereit. Und wie sollte es auch anders sein: Im Vergleich zu Google Photo bietet der Adobe Photoshop Express Editor eine deutlich umfangreichere Fülle an Bearbeitungsmöglichkeiten: So gibt es unter *Edit* neben den verschiedenen Filtern und Effekten auch Werkzeuge, mit denen gezielt Bildbereiche bearbeitet werden können. Ein Stempelwerkzeug, das fehlerhafte Bildstellen passgenau ausbessert, siedelt sich ebenfalls in der Werkzeugleiste an. Unter *Decorate* findet der ambitionierte Heimanwender neben dem Textwerkzeug verschiedene Sprechblasen sowie Cliparts, die auf dem zuvor bearbeiteten Bild frei platziert werden können (Bild 4). ▶



Adobes Browser-Variante von Photoshop bietet neben verschiedenen Filtern auch Sprechblasen und Cliparts (Bild 4)

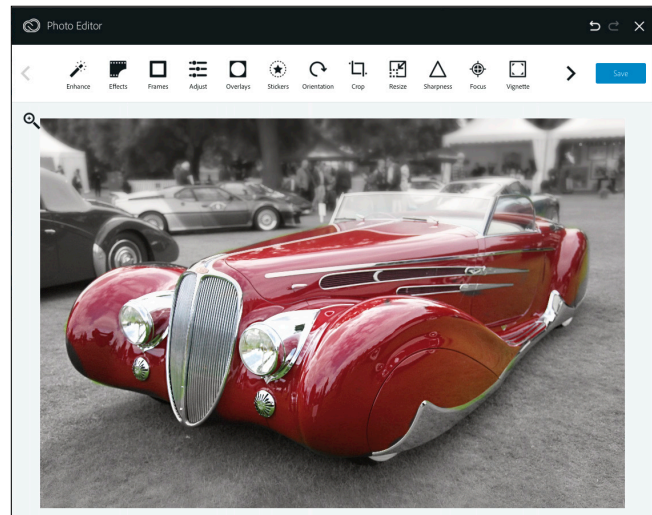
Neben Photoshop Express stellt Adobe den Foto Editor Aviary (www.aviary.com) online bereit, der auch als App für Android oder iOS erhältlich ist.

Den Bildbearbeitungs-Einsteiger erwartet eine sehr übersichtliche Oberfläche, die jedoch eine ganze Menge an Bearbeitungsmöglichkeiten enthält: Eine Leiste an der oberen Kante listet der Reihe nach alle möglichen Bearbeitungsschritte auf – über Verbessern, Effekte und Rahmen bis hin zu den Einstellungen von Helligkeit, Kontrast oder Bildschärfe. Sogar eine Art Light-Version des Stempelwerkzeugs sowie weitere Werkzeuge zum Färben und Aufhellen sind zu finden (Bild 5).

Weitere Online-Editoren: die Profis

Neben Adobe und Google sind im Netz weitere Anbieter vertreten, die sich ganz auf die Online-Bildbearbeitung konzentrieren. Zwei dieser Lösungen zeichnen sich durch einen erheblichen Funktionsumfang aus, Pixlr und Sumopaint:

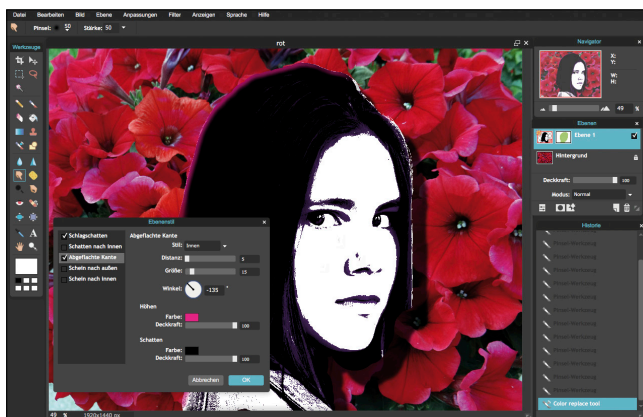
- Pixlr Editor (<http://pixlr.com/editor/>): Nicht nur vorhandene Bilder lassen sich mit dieser kostenlosen Online-Lösung bearbeiten, es können auch neue Bilddateien angelegt werden. Dabei berücksichtigt das Tool auch gängige Bannergrößen. Beim Öffnen eines Bildes erwartet den Anwender eine Oberfläche, die fast ein wenig an Photoshop erinnert: Links liegt eine gut bestückte Werkzeugleiste, rechts neben dem Dokument erscheinen verschiedene frei verschiebbare Paletten. Am oberen Rand liegen die Klappmenüs, darunter befinden sich kontextabhängige Einstellungen zu den einzelnen Werkzeugen. Hier zeigt sich Pixlr spendabel: Zu den Auswahlwerkzeugen gehören ein Lasso und ein Zauberstab, die verschiedenen Zeichenwerkzeuge sind recht gut einzustellen und die Reparaturwerkzeuge reichen gut für Alltagsaufgaben. Die einzelnen Bestandteile des Bildes finden auf unterschiedlichen Ebenen Platz, die wiederum mit einem Ebenenstil versehen werden können. Auch Ebenenmasken stehen bereit, um einzelne Bildbereiche ein- oder auszublenden. Misslingt ein Pinselstrich oder sind mehrere Arbeitsschritte misslungen, hilft die Historie weiter (Bild 6).
- SumoPaint (www.sumopaint.com/): Eine ähnliche Oberfläche, wenn auch optisch nicht ganz so einladend, zeigt Su-



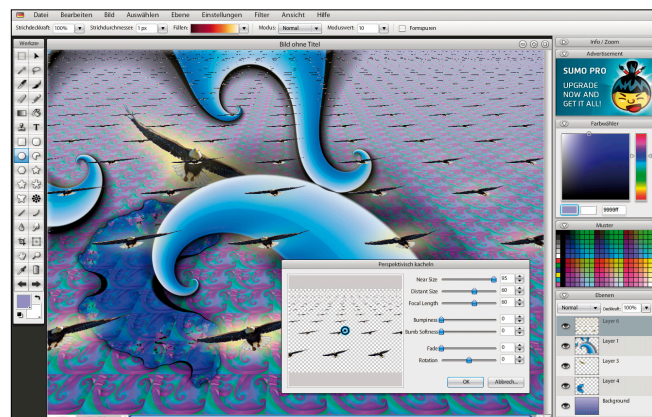
Schnelle Bildbearbeitung online gewährt der Foto-Editor Aviary, ebenfalls von Adobe (Bild 5)

moPaint. Auch hier ist rechts wieder die Ebenen-Palette anzutreffen, eine Historie fehlt jedoch leider. Dafür erweisen sich die Ebenenstile als recht flexibel und das Filter-Menü ist gut bestückt. Hier sind wahre Exoten vertreten, beispielsweise ein Fraktal-Designer, verschiedene Lichteffekte und 3D-Filter wie der Effekt *Perspektivisch kacheln*. Spätestens die Werkzeugleiste sollte dann auch dem größten Skeptiker ein wohlwollendes Schmunzeln entlocken: So stehen etwa beim Verlaufswerkzeug ganze acht Typen zur Auswahl, die wiederum unterschiedliche Modi anbieten. Zu den üblichen Auswahl-, Zeichen- und Bearbeitungswerkzeugen gesellen sich neun zum Teil recht ungewöhnliche Formwerkzeuge. Im Gegensatz zum Editor Pixlr, der zwar Einstellungen per Gradationskurven zulässt, aber nur eine Auto-Tonwertkorrektur bietet, zeigt SumoPaint eine gut justierbare Tonwertkorrektur (Bild 7).

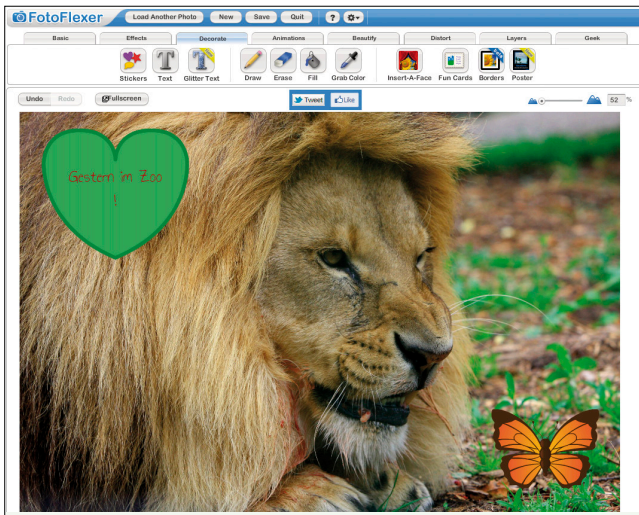
Um schnell die Farben des Urlaubsschnappschusses für den Reiseblog zu korrigieren, ein Bild zurechtzuschneiden oder mit dekorativem Text zu versehen, reicht auch ein weniger aufwendiger Editor.



Pixlr ist ein Online-Editor, der durchaus brauchbare Werkzeuge liefert (Bild 6)



Originelle Tools und Werkzeuge zum Erstellen von Mustern und Hintergründen bietet SumoPaint (Bild 7)



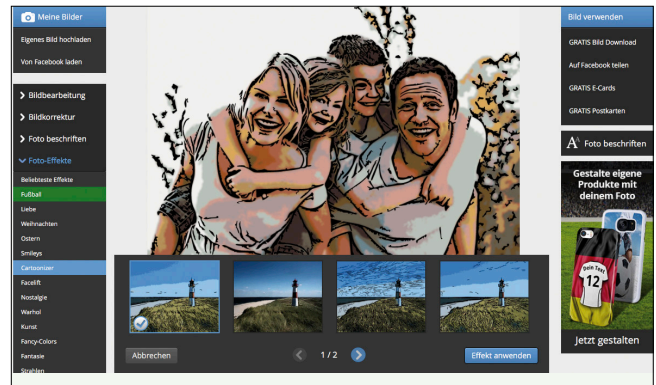
Fotoflexer: Hier liegt der Fokus auf den verschiedenen Spaß-Effekten (Bild 8)

Hier sei zunächst der Fotoflexer genannt. Nicht ganz so schick wie bei den bereits vorgestellten Lösungen verstecken sich hier die Funktionen in einzelnen Reitern, eine Werkzeugleiste gibt es nicht. Zwischen den vielen Auto-Korrekturen und Effekten finden sich dann doch einige einstellbare Filter, etwa zum Einstellen von Sättigung und Licht. Sogar Ebenen für weitere Bildbereiche sind gegeben. Der Fokus der

Die wichtigsten Schritte in der Bildbearbeitung

Nicht jedes Bild zeigt die gleiche Qualität. Kommt das Bild nicht aus Profi-Hand, ist mit folgenden Bearbeitungsschritten zu rechnen:

- **Retusche:** Störende Elemente lassen sich gut mit dem Stempel-Werkzeug entfernen. Dabei wird eine Bildstelle daneben aufgenommen und auf die fehlerhafte Stelle übertragen.
- **Bildrauschen reduzieren:** Wurde das Bild mit einer Kamera mit kleinem Sensor aufgenommen und waren die Lichtverhältnisse nicht ideal, kommt es oft zu einem Bildrauschen. Um diesen störenden Effekt zu mildern, bieten die meisten Lösungen recht gute Rauschreduzierungsfilter.
- **Tonwertkorrektur und Gradationskurven:** Hierbei werden die Helligkeitswerte im Bild umverteilt, um den Kontrast zu regulieren, und eine Farbkorrektur per Kanal wird durchgeführt. Diese Korrektur greift bei flauen oder bei unterbelichteten Bildern sowie bei Fotos mit Farbstich.
- **Sättigung:** Nicht jedes Bild braucht es, doch manche Bilder gewinnen durch eine geringe Anhebung der Sättigung an Intensität. Photoshop CC bietet hierfür die Einstellung Dynamik, die für natürlichere Ergebnisse sorgt.
- **Schärfen:** Grundsätzlich sollte jedes Foto leicht geschärft werden. Erste Wahl ist hier der Filter *Unschärf Maskieren*.
- **Zuschneiden:** Erst wenn alles im Bild stimmt, wird es gemäß der individuellen Vorgaben gestutzt und auf die passende Größe gebracht.



PhotoFancy: Für die ganz schnellen Korrekturen sowie einfache Effekte (Bild 9)

Lösung liegt jedoch deutlich auf den verschiedenen Spaß-Effekten, den teils animierten Cliparts und Text zum Kommentieren des Bildes für Social-Media-Anwendungen (Bild 8).

PhotoFancy wirbt mit 1-Click-Profilkorrektur, tollen Effekten und magischen Namensmotiven – und beschreibt damit den Funktionsumfang recht gut. Alle Funktionen sind in fünf Klappenmenüs am linken Bildschirmrand untergebracht. Zwar gibt es neben der automatischen Korrektur verschiedene Stellschrauben für die elementaren Korrekturen, die sich jedoch ohne Zoom in der viel zu kleinen Bilddarstellung kaum überprüfen lassen. Immerhin gibt's eine recht gute Rote-Augen-Korrektur sowie einige nette Filter (Bild 9).

Fazit

Natürlich ist für professionelle Fotobearbeitung Adobe Photoshop CC die erste Wahl. Soll jedoch das Bildmaterial für das Web oder für mobile Anwendungen bearbeitet werden, spielen Dinge wie einstellbarer Farbraum oder eine hohe Auflösung eher eine untergeordnete Rolle. Muss es dann noch schnell gehen und steht das Profi-Tool eben gerade nicht zur Verfügung, leisten Online-Editoren durchaus gute Arbeit.

Geht es um seriöse Bildbearbeitung, ist Pixlr die beste Wahl, da hier gut einstellbare Filter bereitstehen. Für Muster im 70er-Jahre-Stil ist SumoPaint am besten geeignet. Zudem liefert der Online-Editor durchaus brauchbare Filter für Bildkorrekturen. Beide Bildbearbeiter zeigen einen recht großen Funktionsumfang, sodass Einsteiger doch mit einer gewissen Einarbeitungszeit rechnen müssen. Wer sich mit Bildbearbeitung nicht so gut auskennt, erreicht mit den beiden vorgestellten Editoren von Adobe recht gute Ergebnisse. ■



Katharina Sckommodau

arbeitet als freiberufliche Autorin, Grafikerin und Dozentin, unter anderem für die Akademie der Bayerischen Presse und für Macromedia. Sie veröffentlicht regelmäßig Beiträge in renommierten Fachzeitschriften und verwirklichte mehrere Buchprojekte.



DEVELOPER WEEK 2016 – NACHBERICHT

Volles Haus, volle Köpfe

Informationen satt bei bester Stimmung und vielen interessanten Gesprächen.

Tolle Stimmung vor Ort, ausgezeichnete Keynote-Sprecher, leckerstes Essen und unterhaltsame Specials am Abend: Der Verfasser dieser Zeilen ist der festen Überzeugung, dass all dies ausschließlich auf das von ihm angefertigte Tag-Cloud-Banner zurückzuführen ist. Etwas Voodoo muss sein. Dieser Ansicht können sich seine Teamkollegen verständlicherweise nicht anschließen.

Aber Spaß beiseite. Die Developer Week 2016 lässt sich recht kurz und prägnant mit einem Wort umschreiben: Wow!

Die Developer Week ist die große Konferenz zur Software-Entwicklung in den Bereichen .NET, Web und Mobile in Nürnberg. Einmal im Jahr öffnet sie für vier Tage ihre Pforten und bietet ein prall gefülltes Vortragsprogramm, Abendveranstaltungen, Specials und eine Fachmesse.

»Wissensvermittlung at its best«, könnte das Motto der Konferenz lauten. In nahezu 200 Vorträgen in verschiedenen Formaten konnten die Teilnehmer Wissen tanken – in der Breite und in der Tiefe. Die Breite reichte von .NET mit all seinen Sprachen, Architektur und Application Lifecycle über

Webentwicklung mit ASP.NET, HTML/JavaScript, Angular, Node.js oder PHP bis hin zu den mobilen Technologien für iOS oder Android oder Cross-Plattform. Aber auch Softskills, Cloud Computing oder auch die Hype-Technologien wie Internet of Things oder die HoloLens von Microsoft waren mit Sessions vertreten.

Und in der Tiefe? Bislang gab es auf der Developer Week zwei Formate: Sessions dauerten eine Stunde – für die meisten Themen eine ausreichende Zeit. Für das intensive Einüben einer Technologie hingegen gab es die eintägigen Workshops, die auch dieses Jahr am letzten Tag der Developer Week stattfanden.

Was ist aber mit Themen, die vom Zeitbedarf dazwischen liegen? Die Antwort auf diese Frage waren die DevSessions. Zwei Stunden standen dem Vortragenden hier zur Verfügung. Das Mehr an Zeit gab den Sprechern die Möglichkeit, etwas interaktiver mit den Zuhörern zu agieren, was gern genutzt wurde. Als Themen standen bei den DevSessions zum Beispiel .NET Core, Bidirektionales Messaging mit SignalR,



Grußwort vom Wirtschaftsreferenten Dr. Michael Fraas:
Die Developer Week ist ein wichtiger Baustein für den Wirtschaftsstandort Nürnberg (Bild 1)

Entity Framework, WPF Troubleshooting, Reactive Programming mit RxJS, SQL-Server-Sicherheitskonzepte, Schnellstart mit Angular 2 oder NoSQL mit PostgreSQL zur Auswahl.

Altbekannt und doch immer wieder neu

So weit die nackten Daten zur Konferenz, die in diesem Jahr zum vierten Mal stattfand. »Ein vierjähriges Kind hat laufen gelernt und erforscht die Welt«, fasste es der Wirtschaftsreferent Dr. Michael Fraas (Bild 1) bei seinem Grußwort der Stadt Nürnberg zusammen.

Und doch ist für die Eltern des vierjährigen Kindes jede Sekunde eine Herausforderung. So ging es auch den Eltern der Konferenz, dem Projektteam unter der Leitung von Florian Bender. Zum vierten Mal im NCC Ost der Messe Nürnberg – die Location hatte sich bewährt.

Das Team wusste inzwischen, wo alles ist oder wen man ansprechen muss, wenn zum Beispiel ein Sprecher ein technisches Problem hat.

Dennoch ist jede Developer Week wieder neu. Dieses Jahr war es beispielsweise die Fußball Europameisterschaft, die so viel Zuspruch fand, dass der eigentlich dafür vorgesehene Raum die Leute nicht mehr fassen konnte. Public Viewing wurde deshalb kurzerhand auch in der Referenten-Lounge angeboten.

Keynotes von Jurgen Appelo und Scott Hanselman

Jurgen Appelo (Bild 2) war nach eigenen Worten einmal Entwickler. Nur war das Ergebnis seiner Arbeit so schlecht, dass ihn sein Team zum Manager hochlobte, sagte der Keynote-Sprecher von sich. »Damals wurde ich ein schlechter Manager.« Heute sagen seinen Mitarbeiter über ihn, dass er als Chef »okay« sei – ein Fortschritt. Das hörte sich alles nicht unbedingt danach an, dass Jurgen (ohne ü, da niederländisch) der richtige Keynote-Sprecher war. Doch das Gegenteil war der Fall.

»Viele Manager führen Menschen, als würden sie eine Maschine steuern – etwa nach dem Motto: Entwickler sind Computer mit Haaren und Beinen«, sagte Jurgen. Dabei sollten Manager das System um die Menschen herum managen und nicht die Menschen. Sie sollten die »Freude pflegen«. Mittel dazu gebe es viele und sei es nur, einen Codereview im Freien durchzuführen oder eine Glocke aufzuhängen, die jeder betätigen kann, der etwas Positives mitzuteilen hat.

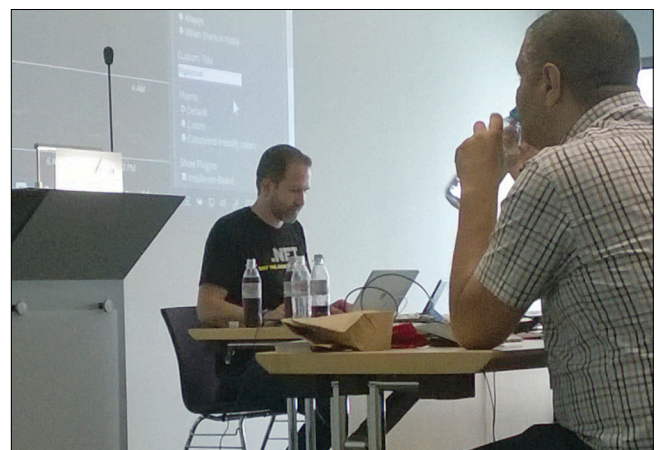
»Nurture happiness« ist einer von sieben Punkten, die Appelo als die Schlüsselkriterien für ein zufriedenes Team sieht. Der Clou ist: Ein zufriedenes Team ist wesentlich produktiver als ein unzufriedenes. Und das ist es ja, was Manager in der Regel erreichen wollen: Produktivität.

Auch Personal Maps können ein Mittel für mehr Nähe und Freundlichkeit innerhalb eines Teams sein. »Ein neues Teammitglied soll sich mit Hilfe eines solchen selbst gestalteten Bildes vorstellen«, sagte er. Eine Personal Map ist wie eine Mindmap mit der Person im Mittelpunkt. Die Art und Weise, wie die Map aussieht, sagt schon viel über die Person aus.

Eine Absage erteilte er finanziellen Boni, die zu einem festgesetzten Termin ausbezahlt werden. Diese würden im ►



Keynote-Sprecher Jurgen Appelo: Manager sollten dafür sorgen, dass ihr Team glücklich ist (Bild 2)



Ganztagesworkshop mit Scott Hanselman: Wie hältst du es mit dem .NET Core? (Bild 3)

schlimmsten Fall sogar Unzufriedenheit säen – beispielsweise wenn die Höhe unpassend sei.

Scott Hanselman (**Bild 3**) hielt einen Ganztagesworkshop und die Closing Session. Der Workshop »ASP.NET Core 1.0 and .NET Core 1.0« war eine Mischung aus knallharten Fakten, Unterhaltung und vielen zusätzlichen Lernschnipseln, die er nebenbei einstreute. Windows 10 beispielsweise unterstützt mehrere Desktops. In der nächsten Version wird man zwischen den Desktops mit einer Vier-Finger-Swipe-Geste hin- und herschalten können.»Und schon fühlt sich Windows an wie ein Mac«, witzelte Scott.

Detailliert erklärte er, warum es in .NET Core momentan eine *project.json*-Datei gibt und warum sie künftig wieder durch eine *csproj*-Datei ersetzt werden wird. »Die JSON-Datei war eine gute Lösung, bis Microsoft Xamarin gekauft hat. Sollen nun verschiedene Projektdateien gemergt werden, lässt sich das über XML-Dateien besser abbilden.«

Er ließ die Teilnehmer auch am Werden von Entscheidungen teilhaben, indem er aus dem Nähkästchen plauderte.

Lang anhaltender Applaus zeigte, dass Scott genau der Richtige für die Closing Session war. Unterhaltsam mit einer großen Portion Selbstironie zeigte Scott (**Bild 4**), warum jeder JavaScript lernen sollte.

Die Cloud, Virtualisierung und JavaScript hätten viel gemeinsam. Die Cloud mache es möglich, mittels eines Sliders Server aufzusetzen. Ein Vorgang, für den man vor der Cloud Stunden oder Tage brauchte. Möglich wird das durch die Virtualisierung. »Dem Betriebssystem wird vorgegaukelt, dass es auf echter Hardware laufen würde, was aber nicht der Fall ist«, sagte Scott.

Auf ihren Maschinen hätten die Anwender aber auch schon jede Menge virtuelle Maschinen laufen. In jedem Browser etwa verarbeitet eine virtuelle Maschine JavaScript.

»JavaScript ist die Assembly Language des Internet.« Um diese Meinung zu unterfüttern, zeigte Scott einen PC-Emulator, der komplett in JavaScript geschrieben ist (<http://bel-lard.org/jslinux>). »Auf meinem Rechner läuft ein Betriebssystem, darauf der Browser, in dessen virtueller Maschine Linux und darin ein Compiler, der hello.c übersetzt«, erklärte Scott.



Keynote mit Scott Hanselman: JavaScript ist die Assembly Language des Internets (**Bild 4**)



Event Storming: Alle machen mit. Learning by doing mit Marco Heimeshoff (**Bild 5**)

Die Sprecher sind das Gold der Konferenz. Sie bringen das Wissen, von dem die Teilnehmer zehren. Und hier konnte die Developer Week 2016 punkten. Rund 200 Experten aus den Bereichen .NET, Web und Mobile, die Erkenntnisse aus ihrem Alltag als Consultant, Entwickler oder Architekt mitbrachten und diese gern weitergaben.

Gepaart mit einem Schuss Unterhaltung wurde auch manch schwere Kost leicht verdaulich. Für Fragen standen alle Sprecher nach ihren Sessions oder in den Pausen zur Verfügung. So konnten die Teilnehmer das Gelernte noch einmal in den Kontext der eigenen Probleme setzen.

200 Experten als Sprecher

Stellvertretend für die Sprecher seien hier ein paar herausgegriffen. Manuel Meyer von Trivadis zeigte, was man mit Visual Studio 2015 so alles machen kann. Welche Arten von Prototypen es gibt und wie man Prototyping machen kann, stellte Veikko Krypczyk vor. Ob auf Papier oder mit Tools wie Balsamiq oder Pencil, spielt prinzipiell keine Rolle. Die Hauptsache ist, dass der Kunde möglichst frühzeitig einen Eindruck erhält, wie die Software aussehen wird und Feedback geben kann.

.NET Core war das Thema einer DevSession mit David Tielke. »NET Core und ASP.NET Core sind cool. Sie ersetzen aber nicht das klassische .NET Framework, sondern bringen eine neue, moderne Herangehensweise in die .NET-Welt. Wir haben uns das Wissen um das .NET Framework mit viel Mühe angeeignet. Microsoft will dieses Wissen bewahren und in eine neue, moderne Welt retten.«

Der große Vorteil sei, dass .NET-Core-Programme nur die .NET-Komponenten enthalten, die das Programm auch tatsächlich braucht, und dass die Programme auf mehreren Plattformen wie Windows, Mac OS und Linux laufen. Momentan sei .NET Core ausschließlich für den Server gedacht. Über Oberflächentechnologien sei noch nichts bekannt. Das bestätigte auch Scott Hanselman in einem Interview.

Angular 2 war das Thema von Manfred Steyer. Ein wichtiger Aspekt in seiner DevSession war der Unterschied zwischen Angular 1 und Angular 2. So gibt es in der Version 2

beispielsweise keinen Scope mehr. »Diese allumfassende Müllhalde, auf die man alles geworfen hat, gibt es nicht mehr«, sagte Manfred Steyer.

Event Storming als eine Methode, um Menschen mit Domainwissen mit den Entwicklern zusammenzubringen, stellte Marco Heimeshoff vor. Allerdings nicht im Frontalunterricht, sondern indem alle Teilnehmer der Session ein Event Storming mitmachten. Tische und Stühle wurden beiseitegeschoben und lange Papierbahnen an den Wänden (Bild 5) mit Post-its beklebt. Auf den Klebezetteln standen alle möglichen Aspekte wie »Ware wurde versandt« oder »Bestellung wurde aufgegeben«. Nach mehreren Runden Sortieren und Clustern kamen automatisch Commands, Queries und Logik heraus, die sich ohne weitere Übersetzung in CQRS (Command Query Responsibility Segregation) und Event Sourcing umsetzen ließen.

#nightone und Community Night

Der Entwickler lebt nicht nur von Wissen allein. Etwas Futter und Getränk gehört auch dazu. Nach der letzten Session gab es am Montagabend wieder die #nightone. Bei einem warmen Abendessen mit Bier, Wein oder alkoholfreien Getränken ließen die Teilnehmer den ersten Tag ausklingen. Wer aber dennoch voller Tatendrang war, konnte in den Night Sessions noch weiterlernen oder einfach nur zugucken.

In der Session zu Ant.Me beispielsweise kämpften virtuelle Ameisenvölker von sechs Teams gegeneinander. Moderator Hendrik Lösch gab den Teams jeweils zehn Minuten, um ihre Ameisen-Klassen zu verbessern und damit einen Überlebensvorteil zu erlangen.

In der Xamarin Night mit Malte Götz, Kai Brummund und Rafael Regh konnten sich die Teilnehmer informieren, wie man mit C# Software für iPhone und Android entwickelt. Trotz unkooperativer Technik vermittelte Mir Aydin Mohammadi, wie wichtig bei der Cross-Plattform-Entwicklung die Organisation der Projektstruktur ist.

Am Dienstagabend waren im Rahmen der Community Night die Pforten auch für Nichtteilnehmer geöffnet. Voraussetzung war eine kostenfreie Registrierung.



Dev-Party bei der Community Night mit Jan Fellien und Constantin Klein (Bild 6)



Die HoloLens stand bei Microsoft im Mittelpunkt (Bild 7)

Dev-Party in Anlehnung an das Spiel Jeopardy hieß das Fragequiz (Bild 6), das Jan Fellien und Constantin Klein mitgebracht hatten. Vier Kandidaten traten gegeneinander an und mussten zu Antworten die Fragen finden.

Nach vier Vorrunden und einem Finale stand der Gewinner fest. Der erhielt ein Ticket für die .NET Developer Conference (www.dotnet-developer-conference.de), die vom 5. bis 7. Dezember 2016 in Köln stattfindet.

Ein Coding-Dojo mit Christian Waha, die Night of Fuckup mit Susanne Mühlbauer und Udo Wiegärtner sowie eine Podiumsdiskussion zum Thema »Universal vs. Native vs. Web« mit Johannes Hoppe, Kerry W. Lothrop, Armin Reuter, Timo Korinth, Marco Richardson und David C. Thömmes waren weitere Angebote der Community Night.

Produkte und Dienstleistungen

Fester Bestandteil der Developer Week ist die Fachmesse, die auf Ebene zwei des NCC Ost stattfand. Hier konnten sich die Teilnehmer über Produkte und Dienstleistungen der Partner informieren. Zu ihnen zählten Microsoft Deutschland (Bild 7), SOS Software Service, MATHEMA Software, Text Control, Fast Reports, AIT, SDX, JetBrains, Developer Express, Saxonia Systems, Heinrich & Reuter Solutions, Telerik, ThingWorx, adorsys, brainLight, GFU Cyrus, SparxSystems Software, User Interface Design, SHS VIVEON, Hemmersbach, Carl Hanser Verlag, ByteAgenten, Senacor Technologies und adesso. Viele Gewinnspiele mit teils hochinteressanten Gewinnen lockten die Teilnehmer an die Stände. Im Büchercafé konnten die Teilnehmer chillen und in Büchern schmökern.

Fazit

Das ging aber schnell – ich meine die Developer Week: Am erschreckendsten bei der Developer Week war die Zeitdilatation. Während die Welt draußen drei Tage verzeichnete, schien die Zeit in den Sälen zehnmal schneller abzulaufen. Doch statt Relativitätstheorie ist eher die Psychologie die Erklärung: Wissensaufnahme, viele Eindrücke, gutes Essen und Gespräche machen die Zeit rasen. Der Verfasser des Artikels freut sich schon unbändig auf die Developer Week 2017 (26. bis 29. Juni 2017).

Tilman Börner ■

STATE OF THE APPLICATION

Statusdaten im Blick

Wie ist der Stand der Dinge in Sachen Zustandsverwaltung bei Applikationen?

Kennen Sie das auch? Anwendungs- und UI-Entwicklung könnte so einfach und entspannend sein, wenn da nicht diese lästigen Daten wären, mit denen man sich ständig herumplagen muss. Fortlaufend ändern die sich. Immerzu muss man die Oberfläche aktualisieren, nur um den aktuellen Status wiederzugeben.

Und dann erst diese Ereignisse und Interaktionen. Alle Nase lang manipuliert jemand am internen Zustand herum und erwartet dann auch noch, daß sich das UI schnell mitändert. Und wehe, man vergisst mal einen Event Handler, schon funktioniert ein nötiger Refresh nicht mehr und der User reagiert völlig zu Recht verschnupft.

Zweifellos sind Daten das Rückgrat jeder Applikation. Schließlich heißt es nicht grundlos Datenverarbeitung. Besagte Daten kommen in vielfältiger Form daher. Um Daten dauerhaft aufzubewahren, werden sie in ein Speichermedium serialisiert. Das kann zum Beispiel eine eigene Datei sein, so wie das Dokument für diesen Artikel. Oder vielleicht eine Handvoll Datensätze in einer Datenbank. In real existierenden Anwendungen gibt es aber noch eine andere Art von Daten, die meist nicht in ein Speichermedium persistiert werden, oder allenfalls nur temporär.

Diese Daten beinhalten alle notwendigen Informationen, die die Applikation intern benötigt, um ihrer Aufgabe nachzukommen und den aktuellen Zustand zu beschreiben. Welcher Menüpunkt ist gerade verfügbar und welcher nicht? Wie ist der Name des aktuell geladenen Dokuments? Welcher Datensatz in der Liste ist gerade selektiert? Hat der Anwender

My Cool App

Lorem	ipsum	dolor sit amet,	consetetur	sadipscing
Lorem i	psum dolor	sit amet, cons	etetur sadipscing el	itr, sed diam
nonumy	eirmod te..	mpor invidunt	ut labore et dolore	magna aliq..
At vero	eos et acc	usam et justo	duo dolores et ea	rebum.
Lorem	ipsum dolor	sit amet, co..	sadipscing elit, sed	diam ...
nonumy	eirmod te..	mpor invidunt	ut labore et dolore	magna aliq..
At vero	eos et acc	usam et justo	duo dolores et ea	rebum.

Dr. Ipsum, Lorem

dolor sit amet
consetetur
sadipscing elit
sed diam
sed diam
nonumy eirmod
tempor invidunt

Labore et dolore
Magna aliquam erat
sed diam voluptua.
At vero eos
et accusam
et justo duo dolores

Save Changes **Cancel**

Struktur: Eine hypothetische Applikation (Bild 1)

das Spaltenlayout verändert oder die Sortierordnung der Tabelle geändert? All diese in der Regel flüchtigen Informationsbröckchen fallen in die Kategorie Statusdaten.

Machen wir ein kurzes Gedankenexperiment. Welche Informationen müsste man sichern, um den Anwender in die Lage zu versetzen, auch in zwei Wochen noch genau an derselben Stelle weiterarbeiten zu können, wenn er jetzt den Computer herunterfährt oder das Browserfenster schließt?

Da wären zunächst die Informationen zum aktuellen Navigationspfad. Eventuell ist auch ein aktives Projekt oder Dokument geladen, das bearbeitet werden soll. Die Position innerhalb des Dokuments beziehungsweise die angezeigten Datensätze wären sicher wichtig.

Hier stellt sich die Frage, ob man tatsächlich Verweise auf die Datensätze speichern sollte, oder doch lieber die Abfrage mit allen Parametern, mit denen diese Auflistung erzeugt wurde? Oder falls es sich doch um ein Dokument handelt: Welche Änderungen hat der Anwender vorgenommen, die noch nicht gespeichert wurden?

Bereits nach kurzem Nachdenken fallen uns für eine vergleichsweise einfache Applikation jede Menge Informationen ein, die nötig wären, um das Benutzererlebnis für das angedachte Feature so perfekt wie möglich zu machen.

Der Status, das unbekannte Wesen

Nicht direkt zum Status gehören aber abgeleitete Informationen, die aus dem aktuellen Kontext stammen. Wenn der Be-

ECMAScript 2015 und Babel

Seit Mitte 2015 ist das neue ECMAScript 2015 beschlossene Sache. Die Browserhersteller haben dies mittlerweile auch mehr oder weniger umgesetzt, bei einigen marktgängigen Browsern aber bislang eher mit Zurückhaltung. Trotzdem muss man auf den neuen, gelegentlich auch als ECMAScript 6 bezeichneten Standard nicht verzichten. Dank Babel ist es möglich, die fortschrittliche Sprachvariante auch heute schon zu benutzen. Hinter den Kulissen übersetzt Babel die neuen Konstrukte in normales ECMAScript 5, das aktuelle Browser in der Regel problemlos ausführen können.

nutzer eingeloggt ist, weiß das System automatisch, welche Sprache er spricht, in welcher Währung er bezahlen wird und ob er als Endkunde Bruttopreise oder als gewerblicher Kunde Nettopreise angezeigt bekommt.

Rein technisch zählt nur die Benutzeranmeldung zum Status, da die anderen Informationen ja aus der tagesaktuellen Benutzerdatenbank stammen. Ob im Hinblick auf unser Benutzererlebnis die Credentials dabei ebenfalls gespeichert werden sollten (natürlich verschlüsselt), oder doch besser nicht, hängt dabei von anderen Faktoren ab. Es spielt für unsere Betrachtung aber auch gar keine Rolle.

Was für eine Desktop-Anwendung bereits eine durchaus interessante Übung ist, wird bei einer mobilen oder Webapplikation noch komplexer. Hier kommt nämlich noch die Frage hinzu, wo diese flüchtigen Informationen über den internen Zustand gespeichert werden. Lebt der Status rein auf der Seite des Clients? Oder gibt es auch Daten, die auf dem Server innerhalb einer Session verwaltet werden?

Kein Patentrezept

Leider gibt es dafür kein Patentrezept. In der Praxis gilt es, einen für den konkreten Anwendungsfall sinnvollen Kompromiss zu finden. Das kann durchaus auch bedeuten, dass Daten rein aus Performancegründen redundant sowohl auf dem Server als auch auf dem Client vorgehalten werden. In bestimmten Fällen ist es aus Sicht einer optimalen Benutzererfahrung sinnvoll, die zusätzliche Komplexität in Kauf zu nehmen, die mit redundanter Datenhaltung unweigerlich einhergeht.

Aufhänger für den Artikel ist die Bedeutung des Status im Kontext der Applikation, insbesondere der Bedienoberfläche. Das scheint zunächst willkürlich. Hält man sich allerdings vor Augen, dass die Oberfläche eines Webservice im Grunde nichts anderes als die von diesem offengelegte Programmierschnittstelle (API) ist, wird klar, dass eine grafische Bedienoberfläche lediglich eine mögliche Repräsentation einer Interaktionsschnittstelle ist. Schließlich spräche nichts dagegen, unserer Anwendung auch eine Kommandozeilenschnittstelle oder ein REST-Interface zu verpassen.

Beschränken wir uns also im Folgenden auf den Status eines GUI. Fürs Erste klicken wir einfach nur ein wenig in unserer hypothetischen Applikation oder Webseite herum (Bild 1).

Während wir das tun, beobachten wir, wie sich aufgrund unserer Interaktion der Zustand ändert. Nach einer Weile stellt sich uns die Frage, woher die Applikation eigentlich weiß, welche Bereiche des UI aufgrund der vorgenommenen Änderungen am inneren Zustand aktualisiert werden müssen? Wieso weiß die Anwendung, dass beim Anklicken eines Eintrags in der Liste die Detailansicht aktualisiert werden muss?

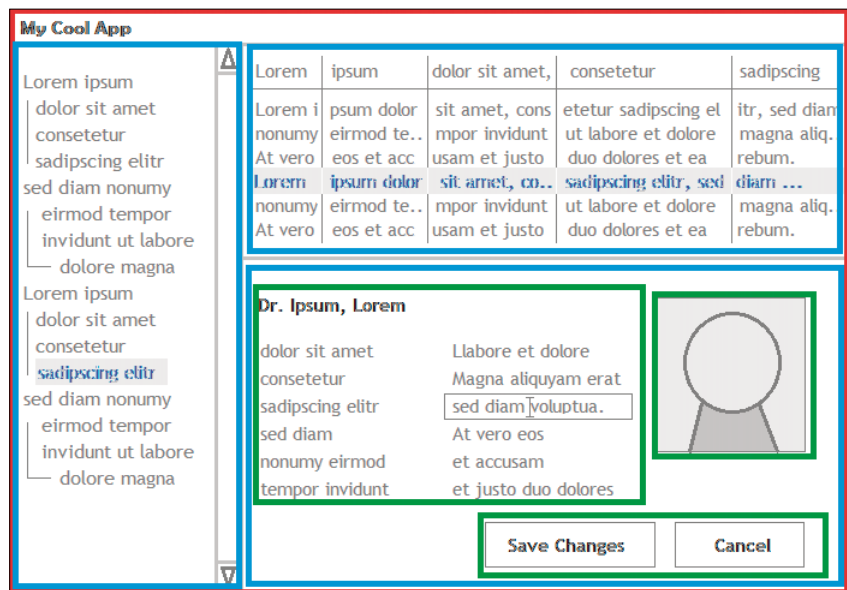
Wer schon das eine oder andere komplexere Formular, ob Web oder Desktop, gebaut hat, der kann davon ein Lied singen. Je mehr Controls und Logik hinter einem UI stecken, desto wichtiger ist eine präzise Planung vor der Implementierung. In der Praxis neigen auch anfänglich kleine und überschaubare Anforderungen dazu, im Lauf der Zeit immer umfangreicher zu werden.

Mit dem Effekt, dass auch aus einem anfangs scheinbar simplen Formular ein komplexes Monster werden kann. Spätestens wenn ein einziger Mausklick wahre Änderungskaskaden auslöst, die sich im UI durch hektisches, sekundenlanges Flackern unzähliger Komponenten widerspiegeln, wissen wir, dass dringend ein Redesign fällig ist (Bild 2).

Zähmung der Widerspenstigen

Angesichts derartiger Probleme und der damit im Quelltext meist einhergehenden Komplexität der Marke gewachsenen System wundert es nicht, dass man sich schon vor Längerem Gedanken über effizientere und besser wartbare Lösungen gemacht hat. Im Lauf der Zeit sind dazu ganz verschiedene Ansätze entstanden.

Die Entwicklung ist aber keineswegs abgeschlossen, auch heute werden bestehende Verfahren verbessert und neue Verfahren aus gewonnenen Erfahrungen und frischen Ideen geboren.



Komponenten: Eine mögliche Aufteilung in Komponenten (Bild 2)

Ihren Siegeszug traten die ersten grafische Oberflächen vor allem auch dank der Objektorientierung an. Ein Mantra der klassischen Objektorientierung ist die Einheit von Daten und Verhalten, die in logischen Einheiten namens Klassen zu kapseln seien. Um die Daten zu manipulieren, werden je nach Lesart und Benennungskonvention entweder Nachrichten an Instanzen der Klasse gesendet, oder aber einfach deren Methoden aufgerufen. Entscheidend ist hier, dass die Daten von außen nicht direkt manipuliert werden können. ►

Also reicht es demnach, einfach die Daten sinnvoll zu strukturieren und dann jede logische Einheit mit Methoden zu versehen? Dass dieser naive Ansatz in der realen Welt wohl doch nicht die letzte Wahrheit sein kann, bemerkt man schnell. Studiert man die bekannten Patterns der Gang of Four, die als praxisrelevante Sammlung gängiger Best Practices auch heute noch brandaktuell sind, wird klar, dass wohl mehr dazugehört.

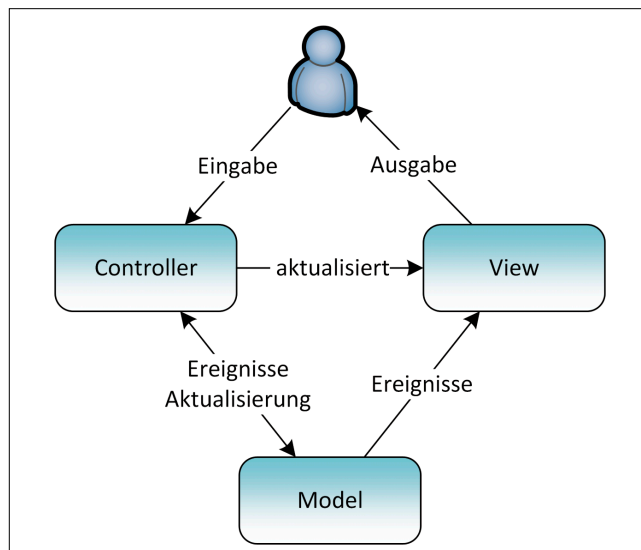
Ein gut aufgebautes System zeichnet sich vor allem durch eine wartungsfreundliche und erweiterungstolerante Struktur aus. Bewährte Mittel sind hier vor allem die Trennung von Verantwortlichkeiten (das bekannte Single Responsibility Principle) und eine sinnvolle und strikte Kapselung. Geringe und wohldefinierte Abhängigkeiten sorgen nicht nur für eine stressfreie Integration eines Modules, einer Bibliothek oder einer Klasse. Sie sorgen auch dafür, dass diese Module isoliert getestet werden können. Als Bonus bekommt man so dann den heiligen Gral der OOP überreicht: Wiederverwendbarkeit.

OOP nur simuliert

Wenn hier von Verantwortlichkeiten die Rede ist, so bezieht sich das nicht nur auf Daten. Gemeint ist vor allem auch die Logik zur Arbeit mit den Daten. Natürlich kann man das Rendering, die Serialisierung in drei Datenformaten, Ausgabeformatierung und fachliche Berechnungen alle zusammen mit den dazugehörigen Daten in eine Klasse packen und sich freuen, wie furchtbar objektorientiert das alles ist. Die schlechte Nachricht? Ist es nicht. Denn hier wird OOP nur simuliert.

Wer kennt sie nicht, diese Formulare, die nicht nur Controls beherbergen, sondern auch jede Menge Logik, um diese anzusteuern? Und natürlich die diversen Statusvariablen, um den Dialog zu steuern. Eine Dialogklasse, sie alle zu regieren ... ein klassisches Brownfield eben, das alle nur denkbaren Nachteile in einer Klasse vereint. Weder Status noch Verhalten sind getrennt testbar. Schlimmer noch, die Wiederverwendung der Formularlogik in einer webbasierten Lösung oder einer Mobil-App ist durch das enge Verweben der drei Aspekte faktisch unmöglich. Da der Code außerdem auch noch eng an die jeweiligen plattformspezifischen Controls gebunden ist, weil es keine Abstraktionsschicht gibt, ist auch nachträgliches Refactoring mindestens arbeitsaufwendig, wenn nicht schon von vornherein zum Scheitern verurteilt.

OOP heißt eben gerade nicht, dass man einfach Daten und Funktionen zusammenpappt und fertig ist die objektorien-



Model-View-Controller: Schematische Darstellung des MVC-Modells (Bild 3)

tierte Laube. Sicher, bei kleinen, überschaubaren Formularen und Anwendungen funktioniert das prima. Unglücklicherweise skaliert das so angeeignete Wissen des Entwicklers aber nicht, sodass komplexere Architekturen gelegentlich trotz vermeintlicher OOP-Expertise dann doch als chaotischer, unwartbarer Klassenverhaue enden.

An dieser Stelle lassen wir die Grundsatzdiskussion, ob OOP nun tot sei oder nicht, einfach links liegen und wenden uns den Erkenntnissen zu, die wir mitnehmen: Offensichtlich ist es also eine clevere Idee, die Bestandteile einer Oberfläche in mindestens drei Verantwortungsbereiche zu trennen. Als Erstes hätten wir den recht offensichtlichen Teil der sichtbaren Controls und sonstigen UI-Elemente.

Außerdem braucht es irgendeine Form von Logik, die für die Steuerung der Abläufe im Dialog, für Datenvalidierung und eventuell nötige Formatierung sowie für die Aufbereitung von Ein- und Ausgabedaten zuständig ist. Idealerweise sind diese Dienstleister selbst komplett statuslos und leichtgewichtig, sodass Erzeugung und Zerstörung jederzeit nach Bedarf erfolgen können. Drittens gibt es natürlich Daten, die sich wiederum in die schon bekannten Gruppen »persistente Anwendungsdaten« und »flüchtiger Programmstatus« aufteilen lassen. Irgendwie müssen wir diese Bausteine nun noch organisieren und miteinander verknüpfen.

Und tatsächlich halten sich nahezu alle heute gängigen Verfahren an dieses grundlegende Schema. Die Begriffe und Aufgaben mögen im konkreten Fall abweichen und Verantwortlichkeiten geringfügig anders strukturiert sein, dennoch läuft letztlich alles auf dieses zugrunde liegende Modell hinaus.

Bäume aus Komponenten

An dieser Stelle noch eine kurze Bemerkung zu den folgenden Beispielen. Es ist natürlich vollkommen klar, dass diese lediglich ein Ausschnitt aus einer Momentaufnahme des aktuellen Status quo der UI-Entwicklung sein können, insbe-

Polyfill – Spachtelmasse für den Browser

Der Begriff Polyfill ist eine Erfindung von Remy Sharp. Im britischen Sprachraum ist Polyfilla ein bekanntes Produkt zum Ausbessern, Füllen und Glätten von Wänden. Als Analogon zur physischen Spachtelmasse ist ein Polyfill für den Webentwickler einfach ein Stück JavaScript, mit dem sich fehlende Funktionen emulieren lassen.

sondere auch ohne jeden Anspruch auf Vollständigkeit. Sie sind vielmehr als Illustration und Anregung zu verstehen, deswegen gehen wir auch absichtlich nicht so sehr auf Implementationsdetails ein. Der gesamte Bereich ist nach wie vor und gerade im Web in heftiger Bewegung, sodass die einzige Konstante tatsächlich die Veränderung ist.

Grafische Oberflächen sind normalerweise so organisiert, dass sich daraus eine baumartige Struktur ergibt. Controls befinden sich in Dialogen oder Anwendungsfenstern und diese wiederum auf einem Desktop mit grafischen Eigenschaften.

Dasselbe wiederholt sich in kleinerem Maßstab innerhalb eines Dialogs, Formulars oder einer Webseite, wo Controls und grafische Elemente in andere eingebettet werden, um eine für den konkreten Anwendungsfall optimal geeignete Struktur zu erzeugen.

Logische Abhängigkeiten

Zu unterscheiden sind hierbei zwei Strukturen. Zum einen die gerade erwähnte visuelle, die sich durch die Schachtelung der UI-Elemente ergibt. Zum anderen gibt es aber auch logische Abhängigkeiten aufgrund der Daten, die diese UI-Elemente repräsentieren. Für gewöhnlich sind beide Aspekte komplett getrennt.

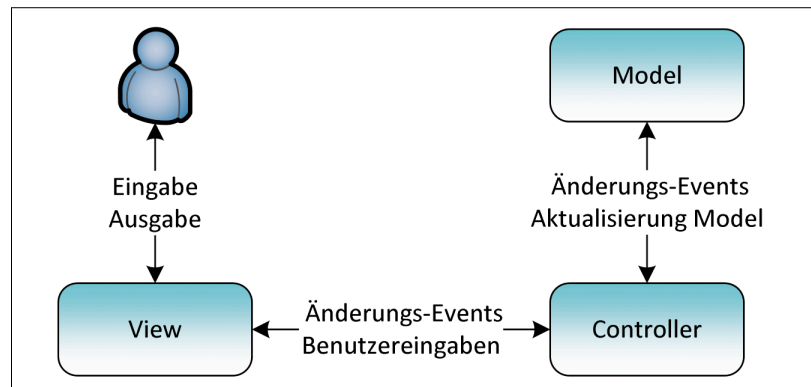
Die bloße Anwesenheit eines bestimmten Datenelements kann dafür sorgen, dass Menüpunkte und Toolbar-Buttons aktiviert werden und sich eine geöffnete Detailansicht mit Daten füllt. Sind in dem Datenelement bestimmte Werte vorhanden, werden in der Oberfläche weitere Daten angezeigt und zusätzliche Optionen und Querverweise sichtbar.

In typischen Webapplikationen sind daher die beiden Belange Layout und dahinterstehende Logik auch sauber getrennt in CSS und JavaScript, das umgebende statische HTML dient oftmals nur noch als Container, der alles zusammenhält. Analoges findet sich bei Oberflächen, die mit XAML gebaut werden. Was den Layouter freut, ist auch für den Entwickler eine tolle Sache. Effektiv kann damit bei geschicktem Aufbau das komplette sichtbare UI wegabstrahiert werden.

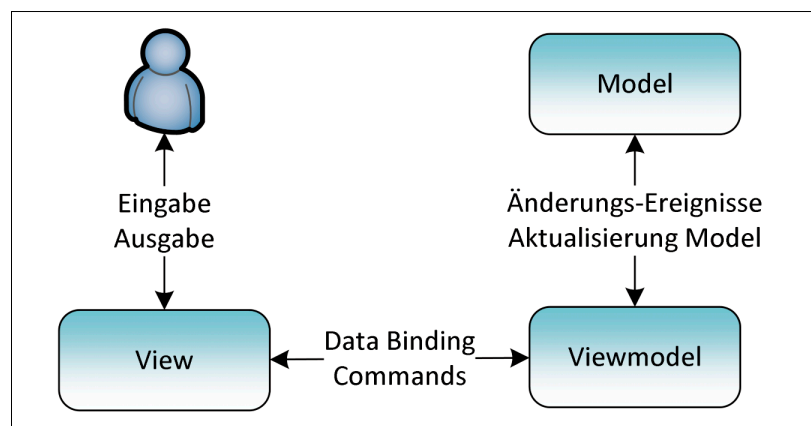
Die Zielstellung ist wieder dieselbe wie oben: Wir möchten eine Trennung von Verantwortlichkeiten, sinnvolle und strikte Kapselung und möglichst geringe, wohldefinierte Abhängigkeiten haben.

Insbesondere das Styling der Oberfläche sollte als separate Verantwortlichkeit unabhängig von der dahinterstehenden Logik sein. Denn letztlich sind sowohl Kunden als auch Anwender an einem konsistenten UI interessiert. Gibt das eine Komponente nicht her, wird sie optisch immer als Fremdkörper auffallen.

Eine wesentliche Aufgabe der Schicht hinter den sichtbaren UI-Komponenten ist es demnach, eine layoutfreundliche Abstraktion der zugrunde liegenden Daten bereitzustellen.



Model-View-Presenter: MVC wurde zu Model-View-Presenter weiterentwickelt (Bild 4)



Model-View-Viewmodel: Eine weitere Variante des MVC-Modells (Bild 5)

Model-View-Konzepte

Der Namensgeber der Gruppe, das Model-View-Controller-Konzept, geht auf Smalltalk-80 zurück. Wie aus dem Namen unschwer herzuleiten ist, gibt es in diesem Modell drei Bestandteile, die miteinander interagieren. Dabei gelten ein paar einschränkende Regeln.

Im klassischen MVC reagiert der Controller auf eingehende Informationen. Diese stammen einerseits vom Model, das bei Datenänderungen Events sendet. Die zweite Quelle für eingehende Ereignisse sind die Interaktionen des Users mit der Anwendung. Der Controller ist verantwortlich für die Aktualisierung von View und Model. Die View kann ihrerseits ebenfalls Ereignisse des Models abonnieren (Bild 3).

Nachteilig an MVC ist, dass auch insbesondere die View noch Logik enthalten kann und wird. Damit ist de facto immer noch eine zu enge Kopplung gegeben. Aus diesem Grund wurde MVC zu Model-View-Presenter weiterentwickelt.

Hauptmerkmal ist hier, dass die gesamte View-Logik mit in den Controller wandert, der dadurch zum Presenter mutiert. Wie vorher schon der Controller, so empfängt auch der Presenter Ereignisse von Model und View und aktualisiert im Gegenzug Model und View. Die View selbst ist jetzt ein rein passives Element, das nur noch die vom Presenter vorformatierten Daten anzeigt und Benutzereingaben entgegen- ►

nimmt. Als weitere Einschränkung kommt hinzu, dass die View keine Verbindung mehr zum Model haben darf, alle Kommunikation verläuft über den Presenter (Bild 4).

Mit dem Ziel, die so gekapselte Logik erstens testbar zu machen und zweitens noch weiter zu abstrahieren und zu standardisieren, sodass sich faktisch beliebige Views an dieselbe Darstellungslogik anschließen lassen, wurde mit Model-View-Viewmodel eine weitere Variante ins Leben gerufen (Bild 5).

Besonders auf XAML-basierten Plattformen hat sich dieser Ansatz als De-facto-Standard durchgesetzt. Kernpunkte des Konzepts sind wieder drei miteinander interagierende Komponenten. Das Viewmodel enthält dabei die Darstellungslogik inklusive Formatierung und Datenkonvertierung. Die Daten werden den View-Komponenten über standardisierte Schnittstellen angeboten und deklarativ über Data Binding angeschlossen. Das Viewmodel verfügt außerdem über eine generische Command-Schnittstelle, die das Auslösen und die Abfrage der Verfügbarkeit von Kommandos ermöglicht. Auch bei MVVM haben View und Model keine direkte Verbindung miteinander.

Ein wesentliches Konzept bei MVVM ist, dass Viewmodels analog zu UI-Komponenten auch geschachtelt eingesetzt werden. Üblicherweise ist ein Viewmodel auch genau für einen Bereich im UI zuständig, sodass die Viewmodels letztlich eine Hierarchie bilden, die den komponentenseitigen Dialogaufbau mehr oder weniger präzise widerspiegelt. Mit MVVM bekommen wir also endlich auch kleine, wohldefinierte Verantwortungsbereiche im UI (Bild 6).

Genau diese kleinteiligen Verantwortungsbereiche verspricht auch HMVC, was für Hierarchisches Model-View-Controller steht. Die Idee hierbei ist, die Struktur der Oberfläche nicht in zwei unabhängigen Bäumen aus Controls und Viewmodels aufzubauen, sondern eine komplette, hierarchische Struktur aus lauter kleinen MVC-Bausteinen zu erzeugen. Die Verbindung der Einzelteile untereinander fällt in den Aufgabenbereich der Controller. Im Gegensatz zu MVVM ist die Oberflächenlogik nicht wirklich gut isoliert und testbar, allerdings eröffnet sich die Möglichkeit, einzelne Bausteine zu entnehmen und diese isoliert zu testen.

Command Query Responsibility Segregation

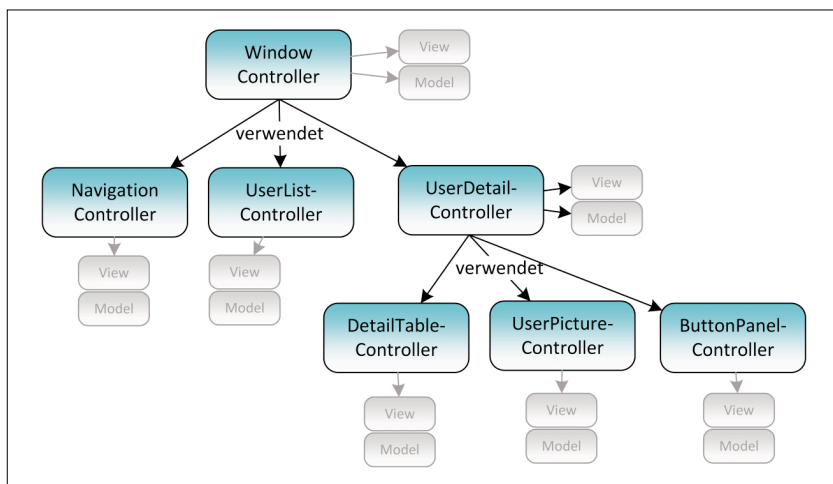
Wir haben uns jetzt eine ganze Weile hauptsächlich mit den oberflächenrelevanten Teilen der Konzepte beschäftigt. Die Frage, wie die Zustandsdaten im Backend manipuliert und abgefragt werden, ist aber mindestens genauso interessant.

Um mit den steigenden Anforderungen an Skalierbarkeit Schritt halten zu können, wurde von Bertrand Meyer ein Konzept namens Command Query Separation (CQS) entwickelt, aus dem wenig später Command Query Responsibility Segregation (CQRS) wurde (Bild 7).

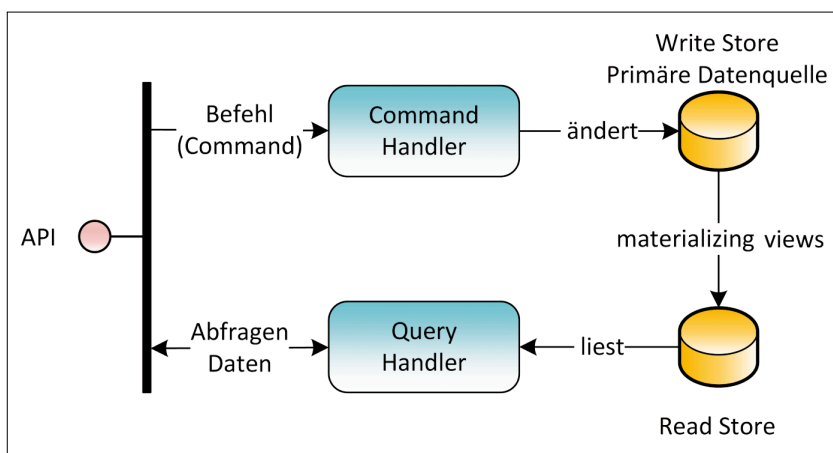
Die Idee hinter beiden ist, dass man reine Datenabfragen von sogenannten Kommandos trennen kann, die tatsächlich Daten manipulieren. CQRS ist daher besonders für Anwendungen geeignet, in denen Lesezugriffe überwiegen. Der Trick hierbei ist, dass reine Lesezugriffe den internen Zustand nicht ändern.

Sie sind komplett frei von Seiteneffekten und lassen sich regelmäßig sehr gut cachen. Solange die zugrunde liegenden Ausgangsdaten unverändert bleiben, kann jede weitere Leseanfrage äußerst effizient bedient werden.

Der Gegenpart zu den rein lesenden Abfragen sind die Kommandos. Das Ändern des Zustands ist ausschließlich durch den Aufruf eines solchen Kommandos möglich. Es liefert nach der reinen Lehre auch keine Rückgabedaten, allerdings ist in der Praxis häufig zumindest eine Quittierung der Abarbeitung erwünscht. Und obwohl das eigentliche CQRS-Pattern im Kern recht sim-



HMVC steht für Hierarchisches Model-View-Controller (Bild 6)



Command Query Separation (CQS): Mit diesem Konzept wollte man mit den steigenden Anforderungen an Skalierbarkeit Schritt halten (Bild 7)

Listing 1: Minimale React-Anwendung

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Hello React!</title>
    <script src="build/react.js"></script>
    <script src="build/react-dom.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/
      libs/babel-core/5.8.23/browser.min.js"></script>
  </head>
  <body>
    <div id="example"></div>
    <script type="text/babel">
      ReactDOM.render(
        <h1>Hello, world!</h1>,
        document.getElementById('example')
      );
    </script>
  </body>
</html>

```

pel daher kommt, ermöglicht die strikte Trennung der Kommandos eine Vielzahl von Wegen zur Implementierung. Die Möglichkeiten reichen hierbei von simpler, synchroner Abarbeitung als Transaktion bis zur verteilten, asynchronen Kommandoausführung via Messagebus oder Event sourcing. Last, not least ist CQRS als generisches Pattern nicht beschränkt auf Server beziehungsweise Backend, sondern kann ebenso auch auf der Clientseite eingesetzt werden.

Nach unserem kurzen Exkurs zur Verwaltung des serverseitigen Status geht es nun wieder zurück an die Oberfläche. Anno 2013 entwickelte Facebook für eigene Zwecke ein Framework namens React, das einen durchaus interessanten Ansatz verfolgt.

Bei React werden nämlich Skriptcode und HTML scheinbar wieder eng miteinander vermischt. Tatsächlich trägt der Schein jedoch, denn das, was auf den ersten Blick aussieht wie in JavaScript eingebettetes HTML, ist nur eine HTML-ähnliche Syntax, die einen Knoten im HTML-DOM beschreibt. Und eigentlich handelt es sich auch gar nicht um normales JavaScript, sondern um JSX.

Strukturen im HTML-DOM

Diese HTML-ähnliche Beschreibung des gewünschten DOM-Teilbaums kann auch Variablen beinhalten, die bei Verwendung innerhalb des Pseudo-HTML in geschweifte Klammern gesetzt werden müssen. React rendert aus diesen Informationen die finalen Strukturen im HTML-DOM selbst, sodass das Ganze plattformunabhängig wird. Und tatsächlich beherrscht React sowohl serverseitiges als auch clientseitiges Rendering. In Listing 1 wird ebenfalls deutlich, wie das Konzept hinter React funktioniert. Jede React-Klasse muss mindestens eine *render()*-Funktion definieren.

Um interne Zustandsdaten zu speichern, sollte darüber hinaus auch eine Funktion *getInitialState()* bereitgestellt werden. Dieser Status ist klassenintern dann einfach über *this.state* abrufbar. Analog wird auf optional in das Element per Attribut hineingereichte Daten über *this.props* zugegriffen. Um den Status zu ändern, ruft man die eingebaute React-Funktion *setState()* auf und übergibt dieser einen kompletten oder auch nur partiellen neuen Status. React wird sodann die aktuellen Daten mit den neuen mischen und anschließend ►

Listing 2: Hello World- Element

```

var World = React.createClass({
  getInitialState: function() {
    return {clicks: 0};
  },

  render: function() {
    return (
      <div className="world" onClick={this.clicked}>
        Hello from {this.props.name}!<0x000A>
        We had {this.state.clicks} visitors so far.
      </div>
    );
  },

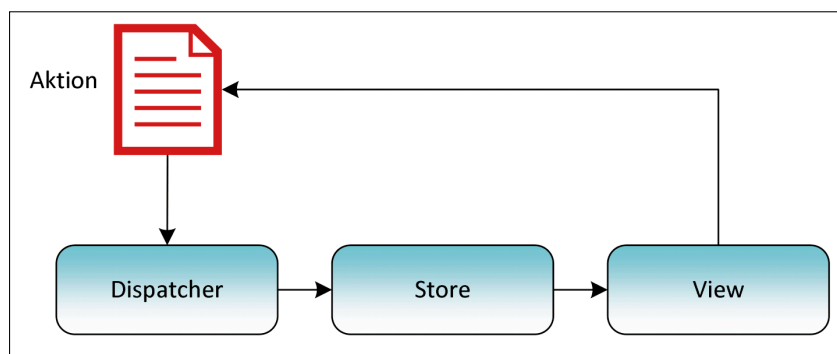
  clicked: function() {
    this.setState({
      clicks: this.state.clicks + 1
    });
  }
});

var MultiHello = React.createClass({
  render: function() {
    return (
      <div className="hello">
        <World name="Earth" />
        <World name="Pluto" />
        <World name="Kepler-452b" />
      </div>
    );
  },

  ReactDOM.render(
    <MultiHello />,
    document.getElementById('content')
  );
});

```

dafür sorgen, dass die `render()`-Funktion aller von Änderungen des Status betroffenen Elemente aufgerufen wird. Das bedeutet nichts anderes, als dass alle betroffenen Teile des DOM neu gerendert werden. Da React aber darauf optimiert ist, tatsächlich nur die geänderten Teile neu zu rendern, geht das vergleichsweise schnell und funktioniert umso besser, je besser sich der Status und das UI in getrennte logische Einheiten aufteilen lassen. Reicht man hingegen immer den kompletten Status durch alle Elemente durch, wird das Konstrukt schnell ineffizient (Listing 2).



Flux: Aufbau der Elemente in Form einer Pipeline aus Actions, Dispatcher, Store und View (Bild 8)

Flux und Redux

So schön das optimierte Rendern bei React funktioniert, so nachteilig ist es auch, dass der interne Zustand viel zu eng mit dem UI verflochten ist. Nicht nur das, darüber hinaus verteilt sich der Status auch noch auf viele kleine Klassen und Ereignis-Handler. Leider tendieren solche Konstruktionen nicht nur dazu, mit zunehmender Größe wartungsunfreundlicher zu werden. Auch die Performance leidet, je mehr reaktive Elemente auf der Seite existieren. Überdies kommt auch noch die potenzielle Gefahr möglicher Inkonsistenzen hinzu, die durch redundante Datenhaltung entstehen kann.

Auch die Entwickler bei Facebook standen irgendwann mit React vor genau diesen Problemen. Um die bröckelnde Ar-

chitektur wieder in den Griff zu bekommen, entstand als Weiterentwicklung Flux. Kernstück von Flux ist der Aufbau der Elemente in Form einer Pipeline aus Actions, Dispatcher, Store und View (Bild 8). Die Statusdaten werden bei Flux nicht mehr in den einzelnen Elementen verstreut vorgehalten, sondern wandern in sogenannte Stores. Damit die Sache auch interaktiv wird, dürfen die Views als einzige Elemente Aktionen erzeugen und in die Pipeline einspeisen. Aktionen sind nichts anderes als Ereignisse. Es sind einfach Nachrichten, die die dahinterliegenden Komponenten darüber informieren, dass sich etwas Bestimmtes am Zustand ändert.

Eine solche Aktion könnte also lauten: »Benutzer hat seine Anschrift geändert« oder »Artikel A mit der Menge B wurde in den Warenkorb gelegt«. Diese Aktionen werden vom Dispatcher an die Stores verteilt. Je nach Anforderungen gibt es meist einen, alternativ auch mehrere Stores pro Applikation. Der oder die Stores sind dafür verantwortlich, den aktuellen Zustand vorzuhalten und zu managen. Abhängig von der eintreffenden Aktion wird der bestehende Status mittels sogenannter Reducer-Funktionen mutiert. Letztere sind im Grunde nichts anderes als spezialisierte Ereignis-Handler. Da die Aktionen streng serialisiert eintreffen, lässt sich der Zustand der Applikation zu jedem Zeitpunkt jederzeit aus dem bekannten Anfangszustand und dem Strom an eintreffenden Aktionen herleiten. Das kann für den Entwickler extrem hilfreich sein, wenn es einen schwer reproduzierbaren Bug zu finden gilt (Listing 3).

Als logische Weiterentwicklung von Flux kann Redux gelten. Genau genommen handelt es sich bei Redux um eine komplett unabhängige Entwicklung, die aber inzwischen als das bessere Flux gehandelt wird. Die aktuelle Version von Redux setzt voll auf ECMAScript 2015 und benötigt eine entsprechend barocke, auf Node.js aufbauende Infrastruktur. Bereits für das einfachste Tutorial-Beispiel schaufelt Node.js weit über 90.000 Dateien auf die Platte. Wie React und Flux lässt sich Redux auch ohne Node.js einsetzen, indem man die an sich recht schlanke JavaScript-Bibliothek manuell einbindet. Allerdings verliert man beim Verzicht auf Node.js auch die Möglichkeit, JSX und ECMAScript 2015 serverseitig zu kompilieren.

Die wichtigste Änderung bei Redux gegenüber Flux ist, dass der Status konsequent als unveränderlich oder immuta-

JSX – Syntaxerweiterungen für JavaScript

Das unter anderem von React verwendete JSX ist eine XML-artige Syntaxerweiterung für JavaScript. Zweck der Übung ist es, mittels der bekannten XML-Syntax hierarchisch gegliederte Strukturen zu beschreiben, die letztlich in HTML-DOM-Konstrukte übersetzt werden. JSX ist an die verworfene EX4-Spezifikation der ECMA angelehnt. Veröffentlicht wurde JSX von Facebook 2014 unter einer Creative-Commons-Lizenz.

Broken-Windows-Theorie

Der Begriff der Broken-Windows-Theorie wurde von den Sozialwissenschaftlern James Wilson und George Kelling geprägt. Er geht auf ein Experiment des Psychologen Philip Zimbardo von 1969 zurück. Im Wesentlichen besagt die Theorie, dass eine sichtbare Beschädigung, im Experiment war dies eine eingeschlagene Fensterscheibe an einem PKW, die Hemmschwelle zu weiterer Zerstörung rapide absenkt. Das Umfeld geht davon aus, daß das Objekt herrenlos ist und sich offensichtlich niemand mehr um dessen Erhaltung kümmert. Besagter Philip Zimbardo studierte gemeinsam mit Stanley Milgram und zeichnete auch für das bekannte Stanford-Prison-Experiment verantwortlich.

Listing 3: Reducer-Funktion

```

var INITIAL_STATE = {
  value: 0,
  evtCount: {
    add: 0,
    set: 0
  }
}

var reducer = function(state, action) {
  if(state === undefined) {
    return INITIAL_STATE;
  }
  var newState = state;
  switch(action.type) {
    case 'add_counter':
      var newEventCount = Object.assign( {},
        state.evtCount,
        {
          add: state.evtCount.add + 1
        }
      );
      var newState = Object.assign( {},
        state,
        {
          value: state.value + action.delta,
          evtCount: newEventCount
        }
      );
      break;

    case 'set_counter_value':
      var newEventCount = Object.assign( {},
        state.evtCount,
        {
          set: state.evtCount.set + 1
        }
      );
      var newState = Object.assign( {},
        state,
        {
          value: action.value,
          evtCount: newEventCount
        }
      );
      break;
  }

  return newState;
}

```

ble angesehen wird und konsequent als Single Immutable State Tree bezeichnet wird. Die logische Folge ist, dass die *reduce()*-Funktionen in Redux nicht mehr den bestehenden Status manipulieren dürfen, sondern wirklich jedes Mal ein neues Statusobjekt für die betroffenen Teilbäume mit den gewünschten Anpassungen erstellen müssen. Diese Einschränkungen haben allerdings zur Folge, dass die Änderungs-erkennung in Redux extrem performant ist, da effektiv nur noch Objektinstanzen verglichen werden müssen

Die finale Lösung?

Ist das nun die finale Lösung aller Probleme? Das sprichwörtliche Ende der Fahnenstange? Keineswegs. Wie mit allen anderen Ansätzen bleiben auch mit Redux ausreichend Probleme übrig. Eines davon betrifft eben genau jenen Single Immutable State Tree, der als einziges Element den Zustand der Applikation verwalten soll. Was in der Theorie großartig klingt, treibt in der Praxis gelegentlich merkwürdige Blüten. der Verzicht auf lokale Statusdaten in den Komponenten bedeutet im Umkehrschluss, dass auch jedes noch so unwichtige temporäre Datenfitzelchen in ebenjenem gemeinsamen Status Tree abgelegt werden muss.

Das aber ist nicht wirklich zielführend, sodass man in der Praxis dann eben doch wieder wie in React zum lokalen Status greift. Das Problem an der Sache ist allerdings, dass die Aufweichung des Gebots »Du sollst deinen Status stets im Single Immutable State Tree speichern« gemäß der Broken-Window-Theorie die Tür öffnet für ungewollte Abweichungen von der reinen Lehre und für Probleme, die redundante Datenhaltung mit sich bringt. Ein sinnvoller Kompromiss

könnte daher sein, dass man vereinbart, solche rein temporären Daten in einem Seitenast des Statusbaums abzulegen.

Fazit

Die Struktur der Daten, die den internen Status abbilden, ist essenziell für jede Anwendung und die Basis für Architektur und Applikationsdesign. Zugleich müssen wir akzeptieren, dass sich die Anforderungen an unsere Applikation im Lauf der Zeit ändern. All diese Änderungen betreffen in irgendeiner Weise die Daten, mit denen die Applikation umgeht. Selbst rein algorithmische Änderungen haben meist Auswirkungen auf das UI und damit wieder auf die zugrunde liegenden Zustandsdaten der Anwendung. Änderungen an den Datenstrukturen sind schwer, aber früher oder später unvermeidlich. Aufgabe von Entwicklern und Software-Architekten ist es, optimale Voraussetzungen zu schaffen, damit die Applikation mit dieser Entwicklung Schritt halten kann. ■



Jens Geyer

entwickelt bei VSX Vogel Software GmbH Lösungen, die bei Kunden weltweit im Einsatz sind. Sein Fokus liegt auf der effizienten Entwicklung hochwertiger paralleler und verteilter Anwendungen. Er ist aktiver Committer/PMC-Member des plattform- und sprachübergreifenden RPC-Frameworks Apache Thrift.

<http://jensgeyer.net>

NÜTZLICHES ZUBEHÖR FÜR DEN RASPBERRY PI

Erweiterungen en masse

Die Basiskomponenten sind beim Single-Board-Computer Raspberry Pi alle auf einem Board. Mit Zubehör kann man aber noch einiges mehr herausholen.

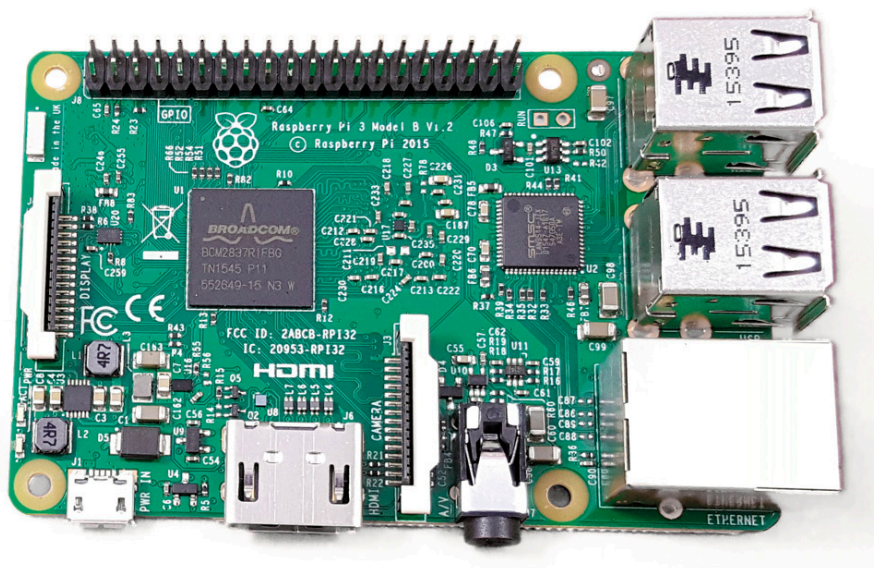
Für knapp 40 Euro bringt die aktuelle Version des Minicomputers, der Raspberry Pi 3 Modell B, auf seiner kleinen Platine so einiges an Leistung mit: ein Quadcore ARM Cortex (Broadcom BCM2837) mit 1,2 GHz und 1 GByte RAM. Dazu kommen einige Anschlüsse für die Kommunikation mit der Außenwelt und zum Anschluss von Zubehör. Die Spannungsversorgung des Minicomputers erfolgt über die vorhandene Micro-USB-Buchse mittels 5 V. Für das erste Raspberry-Pi-Modell wurde ein Netzteil mit 700 mA empfohlen, beim aktuellen Raspberry Pi 3 sind es schon beachtliche 2,5 A. Dies schafft ein herkömmliches Handy-Ladegerät nicht mehr. Inzwischen gibt es auch ein offizielles Netzteil für den Raspberry Pi mit einer maximalen Leistung von 13 W. Das Netzteil bekommt man für 8 Euro, direkt mit EU-Netzstecker.

Wer langfristig und vor allem zuverlässig mit dem Raspberry Pi arbeiten möchte, sollte nicht beim Netzteil sparen. Zwar funktioniert auch ein Handy-Ladegerät, wer aber eine kons-

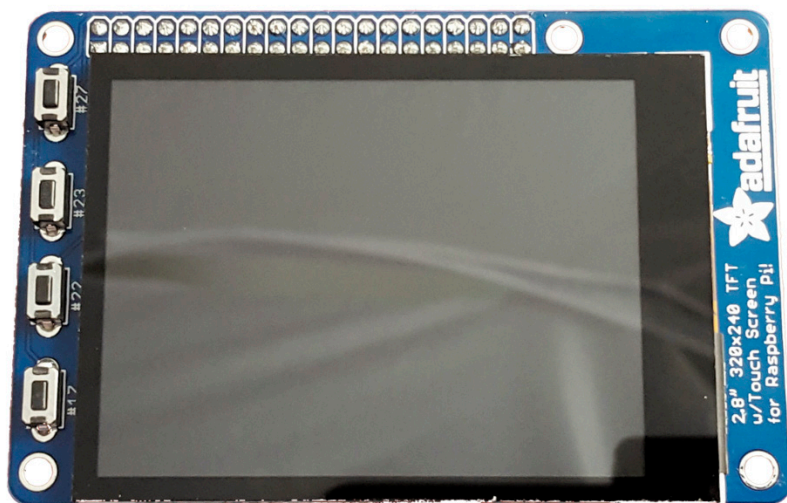
tante Spannung möchte, sollte eher zu einem stabilisierten Netzteil greifen.

Für die Verbindung mit dem Netzwerk stehen eine Ethernet-Buchse (10/100) und ein WLAN-Chip (2,4 GHz, 802.11n) bereit. Im Unterschied zu den Vorgängermodellen spart man sich mit dem neuesten Raspberry-Pi-Modell den externen USB-WLAN-Stick. Zusätzlich bietet der Raspberry Pi 3 auch direkt Bluetooth (4.1 Classic und Bluetooth Low Energy). Darüber kann schon einiges an Zubehör angeschlossen werden (Bild 1).

Zusätzlich stehen insgesamt vier USB-Schnittstellen (Version 2.0) auf der Platine zur Verfügung. Maximal können in Summe 1,2 A über alle vier USB-Schnittstellen bezogen werden. Für stromhungrige Geräte ist das nicht ausreichend, deshalb sollte an diesen Geräten eine externe Spannungsversorgung genutzt werden. Ein Monitor kann entweder über die HDMI-Schnittstelle oder den analogen Composite-Anschluss (kombiniert mit Audio) angeschlossen werden. Für den Anschluss des Raspberry Pi Camera Module steht eine spezielle CSI-Schnittstelle bereit, und ein Display kann über DSI (Display Seri-



Der Raspberry Pi 3 Modell B hat sich in der Größe nicht verändert (Bild 1)



Das Display ID 2423 kommt direkt mit vier aufgelöteten Tastern (Bild 2)

al Interface) angeschlossen werden. Neben diesen Anschlüssen steht noch eine programmierbare GPIO-Schnittstelle (GPIO ist die Abkürzung für General Purpose Input Output) zur Verfügung. Einige Pins der 40-Pin-Leiste sind für Kommunikationsprotokolle vorreserviert. Neben den frei nutzbaren Pins stehen Pins für UART (Universal Asynchronous Receiver Transmitter), I2C (Inter-Integrated Circuit) und SPI (Serial Peripheral Interface) zur Verfügung.

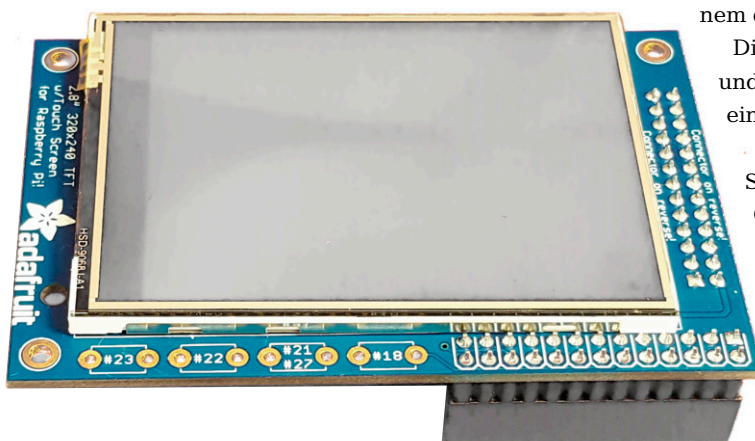
Displays für den Raspberry Pi

Beim Einsatz als Desktopersatz wird man meist einen Monitor an den HDMI- oder Composite-Anschluss anschließen. Wenn es aber um ein Bastelprojekt geht und man trotzdem eine Ausgabe – zum Beispiel als Anzeigebildschirm für einen Roboter oder eine Wetterstation – benötigt, sind andere Lösungen gefragt. Einerseits besteht die Möglichkeit, per Netzwerkverbindung auf den Raspberry Pi zuzugreifen, andererseits kann man auch ein Display an den DSI-Anschluss oder die GPIO-Schnittstelle anschließen. Für die GPIO gibt es etliche Displays von diversen Herstellern. Für diesen Artikel wurden drei unterschiedliche Displays in Betrieb genommen:

- Adafruit PiTFT Plus 320x240 2.8" TFT + Capacitive Touchscreen Mini Kit – Pi 2 and Model A+ / B+, ID: 2423
- Adafruit PiTFT – Assembled 320x240 2.8" TFT+Touchscreen für Raspberry Pi, ID: 1601
- Watterott RPi-Display B+ – 2.8" Touch-Display für Raspberry Pi 1 A+/B+, Pi2 und Pi3, Artikelnummer: 20110952

Die beiden aufgeführten Displays von Adafruit sind in der Installation sehr ähnlich, deshalb wird nur das erste Display in der Liste (ID 2423) näher beschrieben (Bild 2), bei Unterschieden zwischen den beiden Displays wird explizit darauf hingewiesen. Die Displays folgen dem HAT-Standard (HAT = Hardware Attached on Top) und führen die GPIO-Anschlüsse zur weiteren Verwendung heraus.

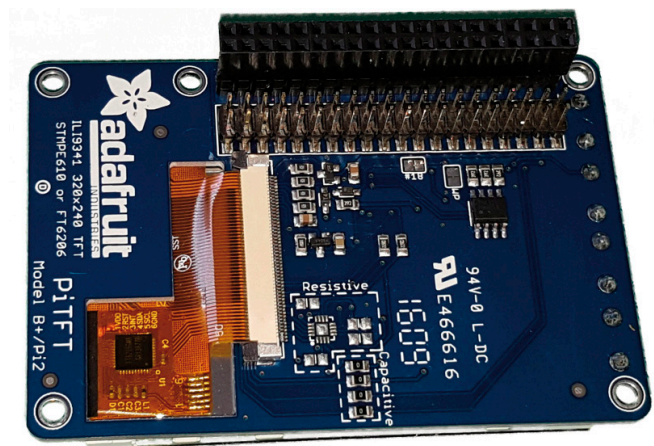
Der größte Unterschied zwischen den beiden Displays liegt in der Art, wie der Touchscreen bedient wird. Das 2423 hat einen kapazitiven Touchscreen, reagiert auf Berührung und kann damit wie ein gängiger Smartphone-Bildschirm durch Fingerstreif bedient werden. Das 1601 hat einen resistiven



Das Display ID 1601 wurde für Raspberry Pi A/B entwickelt, funktioniert aber auch bei den Modellen mit 40 Pins (Bild 3)



Das Display-HAT füllt die komplette Fläche des Raspberry Pi 3 aus (Bild 4)



Über die Rückseite des Displays kommt man an die herausgeführten GPIO-Pins heran und kann diese für andere Projekte nutzen (Bild 5)

Touchscreen und reagiert auf Druck. Damit ist ein Stylus oder Fingernagel erforderlich. Die unterschiedlichen Technologien spiegeln sich auch im Preis wider: Das 2423 bekommt man in Deutschland für circa 48 Euro und das 1601 (Bild 3) für circa 38 Euro. Im Unterschied zum 2423 hat das 1601 keine vier aufgelöteten Taster. Als Smartphone-Nutzer kommt einem die Bedienung des 2423 etwas intuitiver vor.

Die Displays haben eine Auflösung von 320x240 Pixel und eine Größe von 2,8 Zoll. Das 2423 passt optimal auf einen Raspberry Pi ab Modell B+ (Bild 4).

Die Displays nutzen die I2C-Pins (SDA und SDL), SPI-Pins (SCK, MOSI, MISO, CS0) und die beiden GPIO-Pins 24 und 25. Die übrigen Pins stehen für andere Projekte zur Verfügung. Alle 40 GPIO-Pins werden über eine Pinleiste auf der Rückseite herausgeführt (Bild 5), beim 1601 werden natürlich nur die 26 Pins nach außen geführt. An die herausgeführten Pins kommt man nur über ein Flachbandkabel heran.

Der schnellste Weg zu einer erfolgreichen Installation für beide Displays von Adafruit ist die Nutzung eines schon fertigen Images von der ►

Projekt-Website. Das Display wird vor dem Einschalten des Raspberry Pi auf die GPIO gesteckt. Nach dem Anschalten wird das Display hell und nach kurzer Zeit sieht man bereits eine Konsolenausgabe. Auch eine Touchbedienung von LXDE ist mit dem Display möglich. Zusätzlich kann auch einfach ein Bild aus dem Framebuffer angezeigt werden.

Das dritte getestete Display (**Bild 6**) verhält sich ähnlich wie die beiden schon genannten Displays. Auch beim Display von Watterott steht ein fertiges Image zur Verfügung. Nach dem Download und Bespielen einer SD-Karte lässt sich das Display ohne Probleme in Betrieb nehmen.

Eine Alternative zu den HAT-Displays ist das Display der Raspberry Pi Foundation. Im Unterschied zu allen drei bisher gezeigten Displays wird das 7-Zoll-Display nicht an die GPIO angeschlossen, sondern an den DSI-Anschluss. Der Raspberry Pi kommt mit einem DSI-Anschluss, dieser befindet sich am äußeren Rand der Platine – den USB-Anschlüssen gegenüberliegend. Auf der Rückseite des Displays befindet sich eine Anschlussplatine (**Bild 7**). Das Display kann als kompletter Bildschirmersatz genutzt werden. Das kapazitive Touch-Display lässt sich sehr gut bedienen, ein Raspberry-Pi-Tablet könnte damit problemlos gebaut werden.

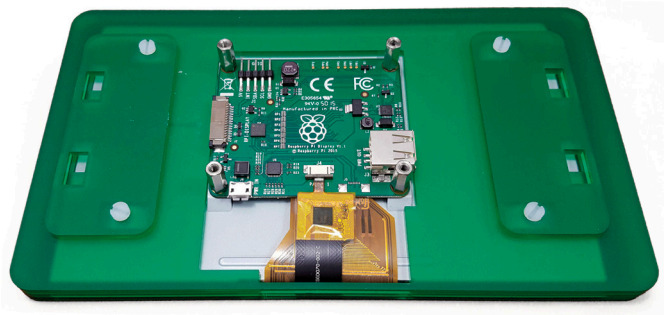
Für das Display gibt es etliche Gehäuse; in diesem Fall wurde ein Gehäuse von Pimoroni genutzt. Nach dem Einsetzen in das Gehäuse wird der Raspberry Pi direkt auf die Platine geschraubt. Über ein Flachbandkabel wird die untere Platine mit dem DSI-Anschluss verbunden. Zusätzlich werden noch zwei GPIO-Pins benötigt: 5 V und Masse (**Bild 8**). Die dafür notwendigen Kabel liegen dem Display bei. Nach der Verkabelung kann das Display direkt unter der neuesten Raspbian Version genutzt werden.

Raspberry-Pi-Kamera

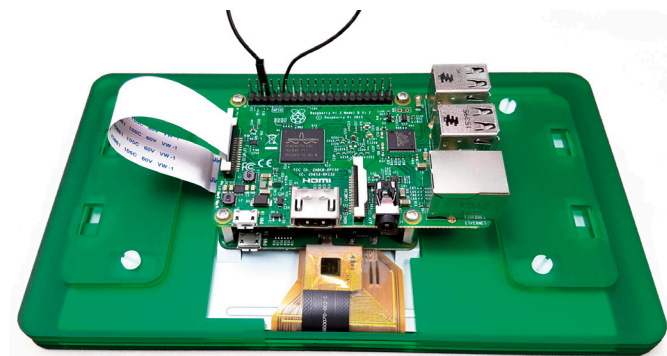
Das Raspberry Pi Camera Module war das erste offizielle Zubehör der Raspberry Pi Foundation und kam im Jahr 2013 auf den Markt. Die erste Version der Kamera – sie wird inzwischen als Version 1.3 verkauft – besitzt eine Auflösung von 5 Megapixel und hat dabei einen OmniVision-OV5647-Sen-



Für knapp 34 Euro bekommt man das 2,8 Zoll große Display im Online-Shop des Herstellers (**Bild 6**)



Gehäuse: Für das Display gibt es unterschiedliche Gehäuse. In diesem Fall wurde das Gehäuse von Pimoroni genutzt (**Bild 7**)



Der Raspberry Pi verschwindet auf der Rückseite des Displays (**Bild 8**)

sor verbaut. Die maximale Auflösung für ein Foto beträgt dabei 2592x1944 Pixel. Die Bilder können in den gängigsten Formaten wie BMP, GIF, JPG und PNG erstellt werden. 1080p-Videos werden mit 30 fps aufgenommen und 720p-Videos mit 60 fps. Die Videos werden in H.264-Codierung aufgezeichnet. 2016 kam dann Version 2 mit 8 Megapixel auf den Markt und hat für die Aufnahmen einen Sony-IMX219-Sensor verbaut, der 8 Megapixel erfassen kann. Die maximale Auflösung für ein Foto beträgt 3280x2464 Pixel. Für ein 1080p-Video werden 30 fps aufgenommen, bei 720p sind es 30 60fps.

Sowohl die erste Version als auch die zweite Version gibt es in zwei Fassungen: eine Version ohne IR-Filter (Produktname: NoIR) und eine Version mit IR-Filter. Bei der Version ohne IR-Filter sind auch Nachtaufnahmen möglich, die Farben bei Aufnahmen mit Tageslicht stimmen aber nicht mit der Realität überein. Das Raspberry Pi Camera Module hat eine Größe von 25x24x9 mm und ein Gewicht von 3 Gramm (**Bild 9**). Preislich liegen die Module bei jeweils circa 30 Euro, das Modul ohne IR-Filter bekommt man teilweise auch fünf Euro günstiger.

Bei älteren Raspbian-Versionen musste die Kamera noch nachträglich integriert werden, beim aktuellen Raspbian Jessie ist keine zusätzliche Installation mehr erforderlich. Für die Nutzung des Zusatzmoduls muss dieses über den CSI-Anschluss mit dem Raspberry Pi verbunden werden. Der Anschluss befindet sich rechts vom HDMI-Anschluss. Für das Einstecken des Flachbandkabels muss der weiße Plastikhalter des CSI-Anschlusses nach oben geschoben werden.

Das Kabel wird dann in den CSI-Anschluss gesteckt und anschließend wird der Halter wieder nach unten geschoben (Bild 10). Der Anschluss des Raspberry Pi Camera Module sollte nur bei ausgeschaltetem Raspberry Pi erfolgen.

Nachdem das Camera Module angeschlossen ist, kann der Raspberry Pi gebootet werden und die Kamera muss nun aktiviert werden. Dies kann man über das zentrale Konfigurationsprogramm *raspi-config* erledigen (Bild 11).

Nun kann die Kamera direkt getestet werden. Am schnellsten testet man die Funktionsfähigkeit über die Kommandozeile. Hier steht das Kommandozeilenprogramm *raspistill* zur Verfügung. Für die Initialisierung ist es wichtig, dass das Bild erst nach einer kurzen Pause von der Kamera aufgenommen wird. Hierfür gibt es den Parameter *-t* oder *--timeout*, gefolgt von der Verzögerung in Millisekunden. Standardmäßig steht dieser Parameter auf 5 Sekunden, empfohlen wird eine Wartezeit von mindestens 2 Sekunden. Der folgende Befehl speichert ein Foto im JPG-Format nach einer Wartezeit von 2 Sekunden im aktuellen Arbeitsverzeichnis ab:

```
raspistill -t 2000 -o bild01.jpg
```

Um das erstellte Bild anzusehen, bietet sich das Kommandozeilenprogramm *gpicview* an. Im Headless-Betrieb muss ein X-Server für die Anzeige gestartet sein. Dem Programm wird lediglich der Name des Bildes als Parameter übergeben. Für die Aufnahme eines Videos steht das Kommandozeilenprogramm *raspivid* zur Verfügung:

```
raspivid -o testvideo.h264
```

Das aufgenommene Video kann man sich über den OMX-Player anzeigen lassen:

```
omxplayer -o hdmi testvideo.h264
```

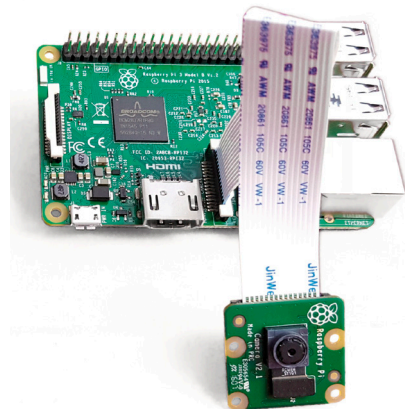
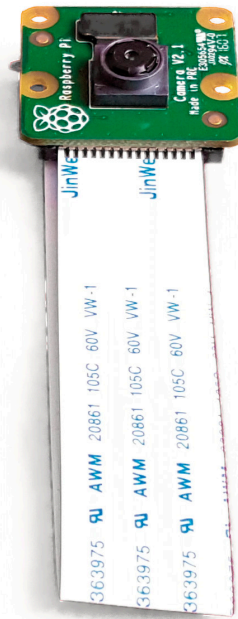
Die Wiedergabe erfolgt aber nur über den Monitor, VNC oder X11-Server funktionieren an dieser Stelle nicht.

Nicht alle Player kommen mit dem H.264-Format zurecht. Eine Konvertierung kann mit MP4Box vorgenommen werden. Dieses Kommandozeilenprogramm muss zuerst installiert werden, dann kann die Konvertierung starten:

```
sudo apt-get install -y gpac
MP4Box -fps 25 -add testvideo.h264
testvideo.mp4
```

Die Kamera kann nicht nur über die Konsole genutzt werden, sondern auch in Python. Hierfür steht eine eigene Bibliothek mit dem Na-

Das Raspberry Pi Camera Module kommt direkt mit Flachbandkabel zum Anschluss an die CSI-Schnittstelle (Bild 9)



Damit das Flachbandkabel sich nicht wieder löst, wird es über den Plastikhalter eingeklemmt (Bild 10)

men PiCamera zur Verfügung. Für die Nutzung der Bibliothek muss diese wie folgt installiert werden:

```
sudo apt-get install python-picamera python3-picamera
```

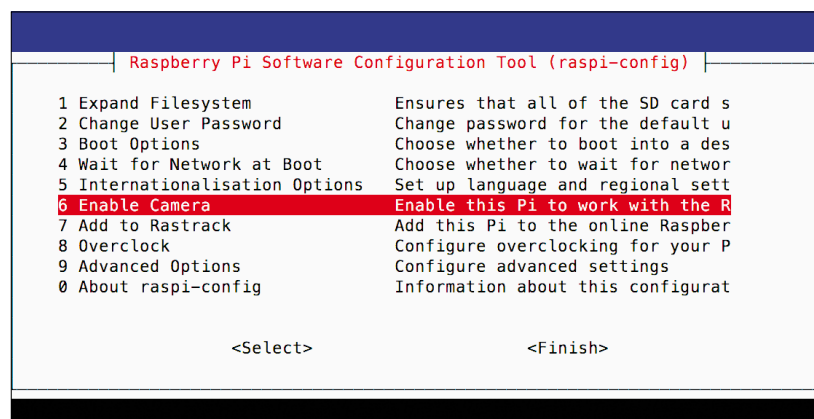
Nun lässt sich die Bibliothek durch Instanziierung von PiCamera in eigenen Skripten nutzen:

```
from time import sleep
from picamera import PiCamera
camera = PiCamera()
camera.resolution = (1024, 768)
sleep(2)
camera.capture('foto.jpg')
```

Inzwischen gibt es das Camera Module auch mit Objektiven.

Festplatte für den Raspberry Pi

Standardmäßig kommt eine SD-Karte als Massenspeicher beim Raspberry Pi zum Einsatz. Große SD-Karten sind ►



Das Camera Module wird über den Menüpunkt *Enable Camera* aktiviert (Bild 11)

aber noch unverhältnismäßig teuer, Alternativen sind deswegen externe Festplatten. Western Digital hat hier eine speziell an den Raspberry Pi angepasste Festplatte auf den Markt gebracht. Die Festplatte trägt den Namen PiDrive und ist in einer Größe von 314 GByte (in Anlehnung an die Kreiszahl Pi) und 1 TByte erhältlich.

Angeschlossen wird die Festplatte an die USB-2.0-Buchse des Raspberry Pi, hierzu benötigt man aber ein Adapterkabel, das der Festplatte nicht beiliegt, sondern separat erworben werden muss. Dies kann man auch direkt im Online-Shop von Western Digital ordern, zusätzlich gibt es auch ein Gehäuse für die Festplatte. Die 1-TByte-Variante gibt es auch als Kit inklusive Adapterkabel. Beim Anschluss der Festplatte wird der Raspberry Pi direkt über das Netzteil des PiDrive mit Spannung versorgt.

Um das Betriebssystem auf die Festplatte zu speichern, ist ein kleiner Umweg erforderlich, denn der Raspberry Pi kann sein Betriebssystem nur von SD-Karte starten. Um trotzdem das Betriebssystem auf die Festplatte zu legen, ist BerryBoot notwendig. Hierbei handelt es sich um einen Bootloader, der auf einer SD-Karte liegt und dann das Betriebssystem von der Festplatte startet. Western Digital stellt auf der Projektseite ein fertig angepasstes BerryBoot als Download zur Verfügung. Das entpackte Archiv muss auf eine SD-Karte aufgespielt werden, und schon kann die Installation starten.

BerryBoot bringt selbst kein Betriebssystem mit; es wird erst während der Installation heruntergeladen. Damit das



Beim Installationsziel kann die angeschlossene Festplatte, hier *sda*, ausgewählt werden (Bild 13)

funktioniert, muss eine Netzwerkverbindung hergestellt werden. Diese wird im ersten Installationsfenster konfiguriert (Bild 12). Im nächsten Schritt definiert man das Ziel der Installation, hier wird dann die Festplatte angegeben (Bild 13). Dann wählt man das Betriebssystem aus einer Liste aus, in diesem Fall wurde Raspbian Jessie ausgewählt. Nach der Auswahl wird das Betriebssystem heruntergeladen (Bild 14) und auf der Festplatte installiert.

Zum Abschluss wird das installierte Betriebssystem als Standard gesetzt (Bild 15). Nach dem Neustart startet dann direkt Raspbian Jessie von der Festplatte. Die SD-Karte muss aber weiterhin im Raspberry Pi bleiben, denn auf ihr befindet sich der Bootloader.

Elektronik an der GPIO

An die herausgeführte GPIO-Schnittstelle mit ihren 40 Pins (ab Raspberry Pi 1 Modell A+/B+, vorher 26 Pins) können elektronische Schaltungen an den Raspberry Pi angeschlossen werden. Wie schon eingangs erwähnt, sind einige Pins für die unterstützten Schnittstellen reserviert (Tabelle 1).

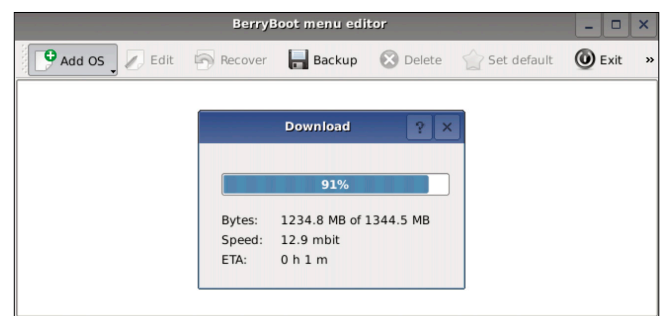
Wer direkt mit der Ansteuerung von Elektronik loslegen möchte, ohne einen eigenen Grundaufbau zu entwickeln, kann auf fertige Aufsteckplatinen zurückgreifen. Für erste Experimente mit LEDs bietet sich das günstige Pi-LITEr von Ciseco an.

Pi-LITEr kann sowohl bei alten Modellen als auch bei den neueren Raspberry-Pi-Modellen genutzt werden. Ab Raspberry Pi 1 Modell B+ bleiben sieben Pinreihen frei. Über die Python-Bibliothek RPi.GPIO lassen sich die Lichter sehr einfach ansteuern.

Wer mehr als nur einfache LEDs steuern möchte, der sollte zu PiFace Digital greifen. Inzwischen liegt diese Aufsteckplatine bereits in einer zweiten Version vor, die ihre Version mit im Namen trägt: PiFace Digital 2. Die Platine bietet acht digitale Ein- und Ausgänge. Zusätzlich sind vier Taster und zwei Relays verbaut. Damit lässt sich schon einiges messen und steuern (Bild 16).



Für die Installation muss ein Betriebssystem heruntergeladen werden, hierfür muss das Netzwerk konfiguriert werden (Bild 12)



Das ausgewählte Betriebssystem wird aus dem Internet heruntergeladen (Bild 14)



Zum Ende der Installation muss das installierte Raspbian Jessie als Standard festgelegt werden (Bild 15)

Die Platine passt optimal auf einen Raspberry Pi und bedeckt komplett die freie Fläche, berührt aber nicht die USB-Buchsen.

Die Ansteuerung erfolgt dabei über SPI. Bei SPI handelt es sich um einen synchronen seriellen Datenbus, der von Motorola entwickelt wurde. Beim Raspberry Pi sind fünf Pins für SPI reserviert.

Die Kommunikation bei SPI erfolgt immer zwischen Master und Slave, wobei der Master die Kommunikation steuert. Der Takt wird über die SCLK-Leitung vorgegeben. Mit den

Tabelle 1: Pins der GPIO-Schnittstelle

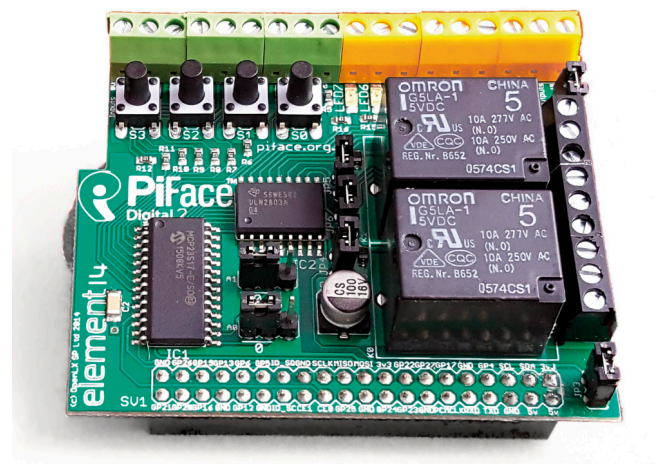
3,3 V	1	2	5 V
GPIO2 (I2C: SDA)	3	4	5 V
GPIO3 (I2C: SCL)	5	6	GND
GPIO4	7	8	GPIO14 (UART: TXD)
GND	9	10	GPIO15 (UART: RXD)
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3,3 V	17	18	GPIO24
GPIO10 (SPI: MOSI)	19	20	GND
GPIO27 (SPI: MISO)	21	22	GPIO25
GPIO22 (SPI: SCLK)	23	24	GPIO8 (SPI: CS0)
GND	25	26	GPIO7 (SPI: CS1)
ID_SD			
I2C: EEPROM	27	28	ID_SC
I2C: EEPROM			
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

zwei Chip-Select-Leitungen (SPI CS0 und SPI CS1) werden die Geräte angesprochen.

Da beim Raspberry Pi nur zwei Chip-Select-Leitungen herausgeführt sind, können damit auch nur zwei Geräte angesprochen werden. Um SPI mit Raspbian Jessie zu nutzen, muss die Schnittstelle zuerst über den Aufruf von *raspi-config* aktiviert werden (Bild 17).

Nach der Aktivierung sollten die beiden Gerätedateien */dev/spidev0.0* und */dev/spidev0.1* vorhanden sein. Bei */dev/spidev0.0* handelt es sich um das Gerät an Chip-Select 0 (CS0) und bei */dev/spidev0.1* um das Gerät an Chip-Select 1 (CS1). Um zu prüfen, ob die Schnittstelle funktioniert, können Sie das Testprogramm *spidev_test.c* nutzen. Dieses müssen Sie zunächst aus GitHub herunterladen und kompilieren.

Über den Aufruf von *spidev_test* können dann Daten über den Bus gesendet werden; damit kann geprüft werden, ob SPI grundsätzlich funktioniert. Sollte der Aufruf nicht erfolgreich gewesen sein, so sollte geprüft werden, ob der Treiber für SPI in der Datei */boot/config.txt* aktiviert ist. Die Zeile *dtoverlay=spi=on* muss dafür ohne Kommentarzeichen enthalten sein. Zusätzlich darf in der */etc/modprobe.d/raspi-blacklist.conf*



Mit PiFace Digital 2 lassen sich vielfältige Steuerungsvarianten einfach und schnell realisieren (Bild 16)

der Eintrag *blacklist spi-bcm2708* nicht enthalten sein. Nach einem Neustart des Raspberry Pi (*sudo reboot*) sollte SPI zur Verfügung stehen. Nach der erfolgreichen Aktivierung von SPI kann PiFace Digital genutzt werden.

Am einfachsten nutzt man die vorhandenen Python-Bibliotheken. Diese können über *apt-get* installiert werden:

```
sudo apt-get install sudo apt-get install
python{,3}-pifacedigitalio
```

Die Installation bringt einige Beispielprogramme mit. Für Python 2 liegen die Beispiele im Verzeichnis */usr/share/doc/python-pifacedigitalio/examples* und für Python 3 liegen die Beispiele im Verzeichnis */usr/share/doc/python3-pifacedigitalio/examples*. An jedem der acht Ausgänge hängt jeweils ►

eine LED. Diese können einfach zum Blinken gebracht werden: Das Beispiel *blink.py* lässt LED 7 im Sekundentakt blinken:

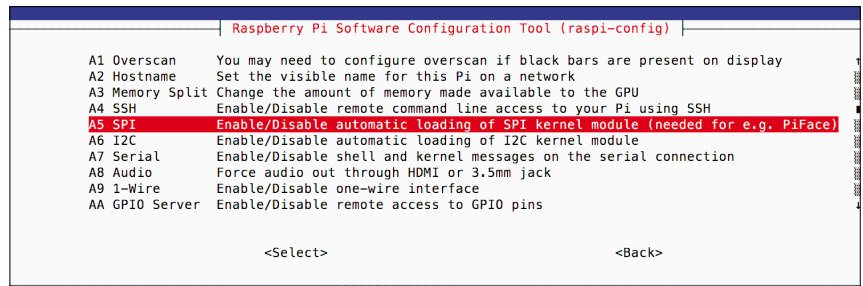
```
from time import sleep
import pifacedigitalio
```

```
DELAY = 1.0 # seconds
```

```
if __name__ == "__main__":
    pifacedigital = pifacedigitalio.PiFaceDigital()
    while True:
        pifacedigital.leds[7].toggle()
        sleep(DELAY)
```

In *presslights.py* sieht man die Arbeit mit den vorhandenen Tastern. Um auf Eingaben zu reagieren, muss ein *InputEventListener* registriert werden. Sobald ein Taster gedrückt wird, wird eine LED angeschaltet. Wird der Schalter losgelassen, dann geht die LED wieder aus.

Die ersten Schritte mit PiFace Digital können durchaus auch komplett ohne Python erfolgen. Hierfür steht eine grafische Oberfläche zur Verfügung, der sogenannte PiFace Di-



Über den Menüpunkt SPI wird der SPI-Bus aktiviert (Bild 17)

gital Emulator. Dieser muss zuerst mit folgendem Aufruf installiert werden:

```
sudo apt-get install python3pifacedigitalemulator
```

Nun kann der Emulator in der Konsole über *pifacedigital-emulator* aufgerufen werden. Im Headless-Betrieb muss dafür ein X11-Server gestartet sein. Nach einer minimalen Wartezeit erscheint die Oberfläche. Diese zeigt das Drücken der Taster an und man kann die LEDs aktivieren. Bei der Aktivierung von Ausgang 0 oder 1 schaltet das jeweilige Relay, dies hört man am Klicken (Bild 18).

Hardware über den I2C-Bus ansprechen

Bei I2C handelt es sich – wie schon bei SPI – um einen synchronen Bus. Entwickelt wurde der Datenbus von Philips Semiconductors im Jahr 1982. I2C ist die Abkürzung für Inter-Integrated Circuit. Seinen Ursprung hat der Bus in der Fernsehetechnik. Hier wurde er für die interne Kommunikation der ICs eingesetzt. I2C ist ein sogenannter Master-Slave-Bus, das heißt, die Kommunikation wird immer durch einen Master initiiert. Dieser Master spricht über eine Adresse einen bestimmten Slave an.

Auch mehrere Master sind möglich. In diesem Fall spricht man dann vom sogenannten Multimaster-Mode. Der serielle Datenbus benötigt nur zwei Leitungen, eine Takt- (SCL) und eine Datenleitung (SDA). Dieser Bus wird auch vom Raspberry Pi an der GPIO unterstützt. Pin 3 (GPIO2) und Pin 5 (GPIO5) sind hierfür vorgesehen. Pin 3 ist SDA und Pin 5 ist SCL.

Damit I2C unter Raspbian Jessie genutzt werden kann, muss dieser Bus über den Aufruf von *raspi-config* aktiviert werden. Zusätzlich muss in der Datei */boot/config.txt* die Zeile *dtparam=i2c_arm=on* hinzugefügt werden. Eigentlich sollte die Zeile bereits vorhanden sein, aber mit dem Kommentarsymbol # versehen sein. Zusätzlich müssen folgende zwei Module in */etc/modules* eingetragen sein: *i2c-bcm2708* und *i2c-dev*.

Jetzt sollten Sie das System neu starten und können schon testen, ob die Module entsprechend geladen sind. Durch Eingabe von *lsmod* sollten Sie prüfen, ob die Module bereitstehen. Um die Funktionsweise zu zeigen, soll der Druck- und Temperatursensor BMP280 von Bosch über I2C in Python angesprochen werden. Für den einfachen Anschluss an die GPIO gibt es mehrere Breakouts auf dem Markt, für diesen Test wurde das Breakout von Adafruit genutzt.

Konfiguration des internen WLAN-Chips

Da der Raspberry Pi 3 schon mit integriertem WLAN-Chip kommt, bietet sich eine Netzwerkverbindung darüber an. Mit der aktuellen Version von Raspbian Jessie (Release-Datum: 2016-05-27) muss keine Zusatzsoftware für die Nutzung von WLAN installiert werden. Über den Kommandozeilenbefehl *ifconfig -a* kann die grundsätzliche Funktionsweise geprüft werden. In der darauffolgenden Ausgabe sollte ein Block mit *wlan0* enthalten sein. Jetzt geht es nur noch um die Nutzung der richtigen Funkkanäle, dafür muss das Land für WLAN im Konfigurationsprogramm *raspi-config* gewählt werden. Im Menüpunkt *Internationalisation Options/Change Wi-fi Country* muss für Deutschland die Option *DE Germany* ausgewählt werden. Nun fehlt noch die Konfiguration von WLAN-Name (SSID) und Passwort in der Datei */etc/wpa_supplicant/wpa_supplicant.conf*:

```
country=DE
ctrl_interface=DIR=/var/run/wpa_supplicant
GROUP=netdev
update_config=1
network={
    ssid="NETZWERK_NAME"
    psk="PASSWORT"
}
```

Mit dieser Konfiguration steht dem Zugriff auf das WLAN nichts mehr im Weg. Lediglich ein Neustart des Raspberry Pi oder ein Aufruf von *sudo service networking restart* ist noch erforderlich.

BerryBoot-Installation über VNC

Die Installation erfolgt entweder über einen angeschlossenen Monitor mit Maus und Tastatur oder über VNC. Für die Installation mit VNC muss das auf SD-Karte aufgespielte Image angepasst werden. Zuerst muss in der Datei *cmdline.txt* eine Zeile mit der IP-Adresse im Format *IP-Adresse/Netzwerkmaske/Gateway* ergänzt werden. Beim Zugriff über WLAN muss zusätzlich der Name des WLAN-Interfaces angegeben werden:

```
vncinstall
ipv4=192.168.0.101/255.255.255.0/192.168.0.254/wlan0
```

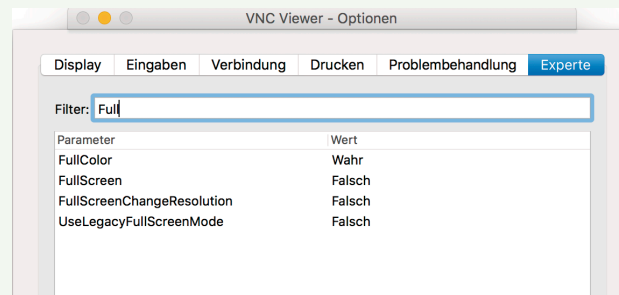
In obiger Zeile wird dem Raspberry Pi die IP-Adresse 192.168.0.101 zugewiesen und als Gateway 192.168.0.254 gesetzt.

Zusätzlich müssen die WLAN-Zugangsdaten in der Datei *wpa_supplicant.conf* hinterlegt werden:

```
ctrl_interface=DIR=/var/run/wpa_supplicant
GROUP=netdev
ap_scan=1
network={
```

```
ssid="WLAN-SSID"
psk="PASSWORT"
}
```

Damit kann das System gestartet werden. Damit man auch ein Bild zu sehen bekommt, muss das Zugriffsprogramm angepasst werden. BerryBoot verlangt nämlich bei der Farbdarstellung 24 Bit. Hierfür muss der Parameter *FullColor* aktiviert werden.



Die Option *FullColor* findet sich in den Experteneinstellungen

Für die Nutzung von I2C auf der Kommandozeile und in Python müssen drei Pakete installiert werden:

```
sudo apt-get update
sudo apt-get install python-smbus python-psutil -y
sudo apt-get install i2c-tools
```

Die Verbindung von BMP280-Breakout und Raspberry Pi erfolgt am besten über Steckbrett und Jumperkabel. Im Beispiel wurde ein Steckbrett vom Typ SYB-46 verwendet. Insgesamt werden vier Verbindungen benötigt: VIN wird mit 5 V verbunden, GND mit Masse, SCK mit Pin 5 und SDI mit Pin 3.

Über das zuvor installierte Kommandozeilen-Tool *ic2detect* (Paket *i2c-tools*) kann der I2C-Bus abgefragt werden. Hier sollte jetzt mindestens eine belegte Adresse angezeigt werden. Sollte der Sensor angezeigt werden, kann dieser nun in Python genutzt werden.

Da man das Rad nicht neu erfinden muss, bietet sich das Adafruit-Beispiel zum BMP280 (<https://github.com/cuda/code/Python-BMP280>) an. In diesem Projekt zeigt das Beispielprogramm *simple-test.py* die Umgebungstemperatur an.

Python-Skript als zentrales Modul installieren

Möchte man die Funktionalität eines Python-Skripts auch in anderen Programmen nutzen, so ist der gängige Weg die Bildung von Funktionen und Klassen und zusätzlich die Einbindung des Skripts in

den Suchpfad von Python. Ein solches Skript wird dann als Modul bezeichnet. Mehrere Module können zu einem Paket zusammengefasst werden. In einem anderen Python-Skript kann das Modul über einen Import (Schlüsselwort *import*) genutzt werden. Der Name des Moduls entspricht dem Dateinamen. Nun muss man nur den passenden Ablageort finden.

Globale Module

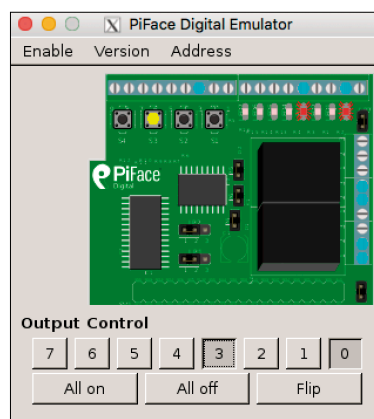
Globale Module und Pakete werden im Verzeichnis *dist-packages* abgelegt. Möchte man ein Modul nur für einen bestimmten Benutzer ablegen, so bietet sich das *site*-Verzeichnis des Benutzers an. Dies ermittelt man wie folgt:

```
$ python -m site --user-site
/home/pi/.local/lib/python2.7/site-packages
```

Unabhängig vom gewählten Verzeichnis, muss man eine *pth*-Datei anlegen und das Modul bekannt machen. In der *pth*-Datei steht lediglich der Name des Moduls. Beispielsweise wird ein *BMP280.pth* und ein Verzeichnis *BMP280* mit darin gehaltener Datei *BMP280.py* in */home/pi/.local/lib/python2.7/site-packages* so angelegt:

```
$ cat BMP280.pth
BMP280
```

Nach erneutem Aufruf von *print-syspath.py* befindet sich das Modul im Suchpfad. Nun kann das Modul aus anderen Python-Programmen heraus genutzt werden. ►



LED 0 und 3 sind aktiviert und Taster S3 ist gedrückt (Bild 18)

Motoren mit dem Raspberry Pi ansteuern

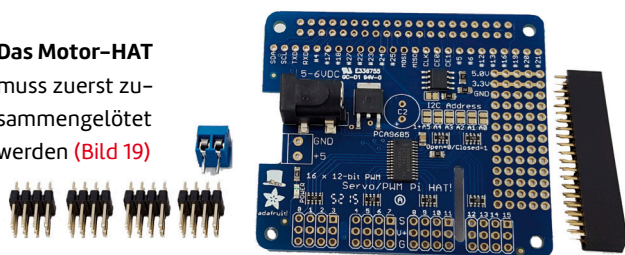
Um Dinge physisch zu bewegen, werden Motoren benötigt. Für dieses Beispiel soll ein Servomotor über den Raspberry Pi angesteuert werden. Für die Ansteuerung eines Servomotors ist ein PWM-Signal erforderlich.

Es gibt bereits fertige Platinen, die die Schaltung für die Ansteuerung der Motoren übernehmen. Auch hier bietet Adafruit wieder gute Lösungen an. Abhängig vom Motortyp gibt es unterschiedliche Platinen. In diesem Beispiel wird das Servo-Motor-HAT genutzt (Bild 19). Über die Platine können insgesamt 16 Servos angesteuert werden. Wichtig ist, dass die Platine über eine separate Spannungsversorgung mit Strom versorgt wird, denn Servos benötigen einiges an Strom.

Nach dem Auflöten der Stiftleisten und des Sockels für das Aufstecken auf die GPIO-Leiste kann das Motor-HAT genutzt werden (Bild 20).

Das HAT wird über den I2C-Bus angesteuert. Nach dem Aufstecken und Starten des Raspberry Pi sollte zuerst geprüft werden, ob der I2C-Bus das HAT gefunden hat. Für die Nutzung der Platine stehen eine fertige Bibliothek und einige Beispiele zur Verfügung. Im Skript *simpletest.py* wird die Ansteuerung eines Servos gezeigt.

Das Motor-HAT muss zuerst zusammengelötet werden (Bild 19)



Vieles gibt es schon fertig

Durch die große Verbreitung des Raspberry Pi ist der Zubehörmarkt inzwischen auch sehr groß und mit zahlreichen interessanten Produkten bevölkert. Eigentlich gibt es nichts, was es nicht gibt. Vor dem Kauf sollte man sich aber unbedingt über Installation und vorhandene Dokumentation erkundigen, bei sehr günstigen Platinen aus Fernost ist fehlende Dokumentation keine Seltenheit.

Hier sollte man lieber dann etwas mehr investieren, um auch die entsprechende Doku zu haben. Als besonders nützlich erweist sich ein USB-TTL-Kabel. Damit kann man auch mal schnell ohne Netzwerk auf den Minicomputer zugreifen.

Für GPIO-Experimente eignet sich Pi-Face Digital 2 sehr gut, wobei es mit seinen knapp 40 Euro schon relativ teuer ist. Attraktiv ist die kompakte Bauweise. Wer aber an Motorensteuerung denkt,



Das Motor-HAT mit angeschloss-nem Servo (Bild 20)

Links zum Thema

- Adafruit
<https://www.adafruit.com>
- Adafruit Servo Motor HAT
<https://www.adafruit.com/products/2327>
- Adafruit Stepper Motor HAT
<https://www.adafruit.com/products/2348>
- BMP 280, Pythonmodul
<https://github.com/cudacode/Python-BMP280>
- Bosch BMP280
https://www.bosch-sensortec.com/bst/products/all_products/bmp280
- PiCamera Dokumentation
<http://picamera.readthedocs.io/>
- PiDrive
<http://wdlabs.wd.com/products>
- PiDrive Dokumentation
<http://wdlabs.wd.com/downloads/#documentation>
- PiFace Digital
www.piface.org.uk/products/piface_digital
- PiGlow
<https://shop.pimoroni.com/products/piglow>
- Pi-LITEr Dokumentation
<http://ciseco.co.uk/docs/Pi-LITEr-v-0-1.pdf>
- Python 2.7 Dokumentation
<https://docs.python.org/2>
- Raspberry Pi
<https://raspberrypi.org>
- RPi.GPIO
<https://pypi.python.org/pypi/RPi.GPIO>
- Technische Daten HS-311
https://www.servocity.com/html/hs-311_standard.html
- Wiring Pi
<http://wiringpi.com>

sollte eher zu einem Motor-HAT greifen, denn PWM liefert PiFace Digital 2 nicht.

Wer den Raspberry Pi viel nutzt, der sollte über die Investition in das 7-Zoll-Touch-Display nachdenken. Die Größe dieses Displays ist sehr angenehm und die Bedienung hervorragend. Insgesamt lässt sich sagen, dass man mit Zubehör die Funktionalität des Minicomputers noch ordentlich erweitert bekommt, dann aber auch preislich weit über den 40 Euro des Raspberry Pi liegt. ■



Dr. Markus Stäuble

ist Informatiker, Conference Chair der Developer Week und Programmleiter Make beim Franzis Verlag. Neben Make beschäftigt er sich viel mit dem Thema Mobile und hat zu dessen Auswirkungen auf die Arbeitswelt promoviert.

<https://github.com/mstaeuble>

Impressum

Verlag

Neue Mediengesellschaft Ulm mbH
Bayerstraße 16a,
80335 München
Telefon: (089) 741 17-0,
Fax: (089) 741 17-101
(ist zugleich Anschrift aller
Verantwortlichen)

Herausgeber

Dr. Günther Götz

Chefredakteur

Max Bold
– verantwortlich für
den redaktionellen Teil –
E-Mail: redaktion@webundmobile.de

Schlussredaktion

Ernst Altmannshofer

Redaktionelle Mitarbeit

Philip Ackermann, Christian Bleske,
Tilman Börner, Ekkehard Gentz,
Jens Geyer, Thomas Hafen, Tam Hanna,
Anna Kobylinska, Bernhard Lauer,
Filipe Martins, Florence Maurice,
Holger Reibold, Michael Rohrlisch,
Sumit Sarkar, Markus Schraudolph,
Katharina Sckommodau, Thomas Sillmann,
Frank Simon, Markus Stäuble

Art Directorin

Maria-Luise Sailer

Grafik & Bildredaktion

Alfred Agatz, Dagmar Breitenbach,
Verena Greimel, Hedi Hefele,
Manuela Keller, Simone Köhnke,
Cornelia Pflanzner, Karoly Pokuta,
Petra Reichensperner, Ilka Rütther,
Sebastian Scharnagl, Christian Schumacher,
Nicole Üblacker, Mathias Vietmeier

Anzeigenberatung

Jens Schmidtman, Anzeigenleiter
Klaus Ahlering, Senior Sales Manager
Telefon: (089) 741 17-125
Fax: (089) 741 17-269
E-Mail Anzeigenberatung: sales@nmg.de

Anzeigendisposition

Dr. Jürgen Bossmann
Telefon: (089) 741 17-281
Fax: (089) 741 17-269
E-Mail: sales@nmg.de

Leitung Herstellung/Vertrieb

Thomas Heydn
Telefon: (089) 741 17-111
E-Mail: thomas.heydn@nmg.de

Leserservice

Hotline: (089) 741 17-205
Fax: (089) 741 17-101
E-Mail: leserservice@nmg.de

Kooperationen

Denis Motzko
Telefon: (089) 741 17-116
E-Mail: kooperationen@nmg.de

Druck

L.N. Schaffrath Druckmedien
Marktweg 42-50
47608 Geldern

Vertrieb

Axel Springer Vertriebsservice GmbH
Objektvertriebsleitung Lothar Kosbü
Süderstraße 77
20097 Hamburg
Telefon: (040) 34724857

Bezugspreise

web & mobile developer ist das
Profi-Magazin für Web- und Mobile-
Entwickler und erscheint zwölfmal im Jahr.
Der Bezugszeitraum für Abonnenten ist
jeweils ein Jahr. Der Bezugspreis
im Abonnement beträgt 76,20 Euro
inklusive Versand und Mehrwertsteuer
im Halbjahr, der Preis für ein Einzelheft
14,95 Euro. Der Jahresbezugspreis beträgt
damit 152,40 Euro.

In Österreich sowie im übrigen Ausland
kostet das Abonnement 83,70 Euro im
Halbjahr. Der Jahresbezugspreis beträgt
somit 167,40 Euro. In der Schweiz kostet
das Abonnement 152,00 Franken im
Halbjahr. Der Jahresbezugspreis in der
Schweiz beträgt 304,00 Franken.

Das Abonnement verlängert sich
automatisch um ein Jahr, wenn es nicht
sechs Wochen vor Ablauf der Bezugszeit
schriftlich beim Verlag gekündigt wird.
Studenten erhalten bei Vorlage eines
Nachweises einen Rabatt von 50 Prozent.

ISSN: 2194-4105

© 2016 Neue Mediengesellschaft Ulm mbH

Jetzt Ihre
web & mobile developer
auf dem iPad lesen



Jetzt online
weiterbilden!

„Moderne Probleme
fordern modernes
Wissen. Mit Webinaren
bleibt man auf
dem neusten Stand.“

Johannes Hofmeister
Softwareentwickler,
Psychologe, Sprecher



developer-media.de/webinare

ABMAHNUNGEN

Latente Gefahr

Hoffentlich nur aus der Theorie bekannt: Abmahnschreiben.

Webshop-Betreiber, Blogger, Social Media Junkies – sie alle leben mit der latenten Gefahr, abgemahnt zu werden. Insbesondere Gewerbetreibende sollten dieses Risiko nicht auf die leichte Schulter nehmen. Das bedeutet, dass man sich mental darauf einstellen sollte, dass es einen irgendwann einmal treffen kann.

Grundlagen

Die Abmahnung ist eine Aufforderung, eine tatsächliche oder vermutete Rechtsverletzung zu unterlassen. Sie ist ein gesetzlich vorgesehenes Instrument zur außergerichtlichen und zugleich verbindlichen Klärung eines Rechtsstreits. Sie dient folglich dazu, eine Streitigkeit ohne Anrufung eines Gerichts, aber dennoch für alle Beteiligten verbindlich zu regeln. Erst wenn dies scheitert, kann dann der Rechtsweg beschritten werden. Dies ist jedoch die zeitlich und finanziell aufwendigere Alternative.

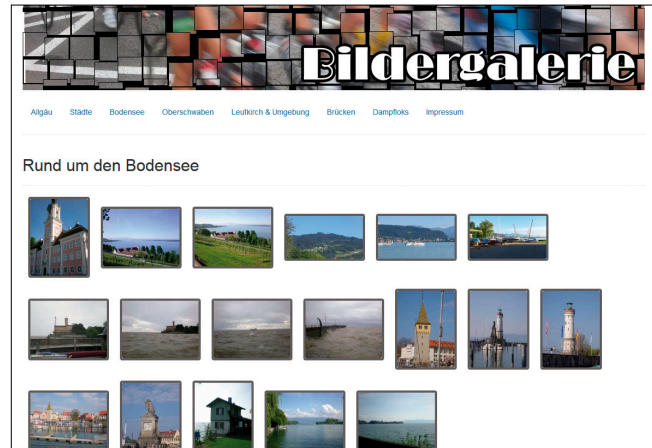
Eine Abmahnung kann prinzipiell in unterschiedlichen Rechtsbereichen erfolgen, sie ist nicht auf die neuen Medien beschränkt. Es gibt sie unter anderem in Bezug auf Content (Urheberrecht), unlauteres Verhalten (Wettbewerbsrecht), geschützte Marken, Patente, Designs (Markenrecht), Versäumnisse im Bereich Datenschutz/IT-Sicherheit (Datenschutzrecht), Beleidigung, Bedrohung oder auch rechtswidriges Verhalten von Arbeitnehmern (Arbeitsrecht).

Abmahnschreiben werden in aller Regel von Rechtsanwälten verfasst. Dies ist zwar sehr sinnvoll, allerdings keine Verpflichtung. So kann beispielsweise der Inhaber von Urheberrechten einen Dritten selbst abmahnen, wenn dieser ungefragt sein Werk nutzt. Allerdings ist eine solche Vorgehensweise nicht unbedingt ratsam, da es ja schon Sinn ergibt, Juristen ihren Job machen zu lassen (Bild 1).

Apropos Abmahnschreiben: Es nicht so, dass eine Abmahnung nur in Schriftform wirksam ist. Auch eine mündlich ausgesprochene Abmahnung ist zulässig – allerdings im Zweifelsfall schlechter nachzuweisen. Der Abmahner muss den Zugang seines Abmahnschreibens beim Abgemahnten nicht nachweisen. Er muss lediglich dezidiert darlegen, dass er die Abmahnung an die richtige Adresse verschickt hat.

Unterlassungserklärung

Mit der Abmahnung wird also das Unterlassen einer bestimmten Handlung verlangt. Folglich ist fast jedem Abmahnschreiben eine vorformulierte Unterlassungserklärung beigelegt, die der Abgemahnte unterzeichnen soll. Die Pflicht zur Abgabe einer solchen Unterlassungserklärung besteht im Fall einer berechtigten Abmahnung zwar, um die sogenannte Wiederholungsgefahr aus der Welt zu schaffen. Geschieht



Die Benutzung von Content wie Bilder ohne Erlaubnis des Rechteinhabers bilden häufig Anlass für eine Abmahnung (Bild 1)

dies aber nicht, so bleibt dem Abmahnenden kaum eine andere Wahl, als den Gang vor Gericht anzutreten.

Es muss jedoch nicht zwingend die vorgefertigte Erklärung akzeptiert werden, denn bisweilen schießen diese ein wenig über das Ziel hinaus. Daher ist es stets ratsam, sich die Unterlassungserklärung sehr genau anzusehen und diese im Zweifel zu modifizieren oder eine gänzlich neue aufzusetzen. Aber Achtung: Hierbei sollte tunlichst ein auf das jeweilige Rechts-

Praxis-Tipp

Anhand folgender Checkliste lässt sich ablesen, was die geeigneten Verhaltensmaßregeln im Fall einer Abmahnung sind:

- Vorfeld: juristische Prüfung der eigenen Website / Social-Media-Profil.
- Im Fall des Falles: Ruhe bewahren.
- Sofortmaßnahme: Prüfung des vorgeworfenen Verhaltens, Notieren von Eingangsdatum und Frist.
- Abmahnberechtigung (Arbeitgeber, Konkurrent, Rechteinhaber, Verbraucherorganisation).
- Frist: angemessene Zeit zur Reaktion? (gegebenenfalls Bitte um Fristverlängerung).
- Streitwert: überhöhte Forderungen? (Anwaltskosten, Schadensersatz et cetera).
- Gegebenenfalls Anzeichen für Rechtsmissbrauch.
- Zweifelsregelung: Rat / Vertretung durch Anwalt oder Verbraucherzentrale.

gebiet spezialisierter Anwalt hinzugezogen werden, um sich nicht ins eigene Knie zu schießen. Denn man muss wissen, dass die Abgabe einer Unterlassungserklärung nichts anderes ist als ein Vertrag, den man mit der Gegenseite eingeht. Und als solcher entfaltet sie ihre Wirkung für 30 Jahre.

Neben dem Versprechen, den abgemahnten Verstoß zukünftig zu unterlassen, muss die Unterlassungserklärung auch strafbewehrt sein. Sie muss also eine sogenannte Vertragsstrafenregelung enthalten, mit der ein bestimmter Betrag festgelegt wird für den Fall, dass es erneut zu besagtem Verstoß kommt. Fehlt eine solche Klausel, so kann die Unterlassungserklärung zwar generell wirksam sein, lässt jedoch nicht die Wiederholungsgefahr entfallen und wird somit faktisch wertlos. Allerdings muss nicht zwingend eine feste Summe festgelegt werden (oftmals sind es über 5000 Euro), es kann auch der sogenannte »Hamburger Brauch« gewählt werden. Dieser nennt gerade keinen genauen Geldbetrag, sondern spricht von einem angemessenen Betrag, dessen Höhe im Zweifel von einem Gericht überprüft werden kann.

Kerngleiche Handlungen

Es gilt darauf zu achten, dass nicht nur der konkret in der Unterlassungserklärung bezeichnete Verstoß zu unterlassen ist, sondern auch die sogenannten kerngleichen Handlungen. Dies hat nicht nur das Oberlandesgericht (OLG) Frankfurt a. M. mit Beschluss vom 26. April 2016 (Aktenzeichen: 6 W 3/16) so gesehen. Ein gegen die konkrete Verletzungsform gerichteter Unterlassungstitel erfasst darüber hinaus auch im Kern gleichartige Abwandlungen, in denen das Charakteristische der konkreten Verletzungsform zum Ausdruck kommt, so die Frankfurter Richter.

Rechtsverstoß, Abmahnung, Unterlassungserklärung – und dann? Wer meint, dass mit der fristgerechten Abgabe der Unterlassungserklärung und der Zahlung der Abmahnkosten alles für ihn erledigt sei, der irrt. Und dieser Irrtum kann unter Umständen auch teuer werden.

Wird man beispielsweise wegen unerlaubter Nutzung eines fremden Fotos auf der eigenen Website abgemahnt, so ist es nicht damit getan, dieses Foto zu löschen, um seinen Verpflichtungen aus der Unterlassungserklärung gerecht zu werden. Es muss nämlich dafür gesorgt werden, dass die betreffende Fotografie nicht nur von der Website verschwindet, also sie auf keiner Site mehr eingebunden ist und auch keine Links mehr auf sie verweisen.

Was ebenfalls gerne vergessen wird, ist die Entfernung des Fotos aus fremden Internetangeboten, also zum Beispiel einer Auktionsbeschreibung bei Ebay, dem Google-Cache, den Google-Suchergebnislisten et cetera. Zwar verneint das OLG Zweibrücken eine solche Pflicht (Urteil vom 19. Mai 2016, Aktenzeichen: 4 U 45/15), hierbei dürfte es sich allerdings um eine Ausnahmeentscheidung handeln. Denn zahlreiche andere Gerichte, unter anderem das OLG Celle (Urteil vom 29. Januar 2015, Aktenzeichen: 13 U 58/14) oder auch das OLG Düsseldorf (Urteil vom 3. September 2015, Aktenzeichen: I-15 U 119/14), gehen von einem Verstoß gegen die Unterlassungserklärung aus. Dies gilt jedenfalls dann, wenn nicht dafür Sorge getragen wird, dass die gängigen, also die bekannt-

Links zum Thema

- Video-Trainings des Autors
www.video2brain.com/de/trainer/michael-rohrlich
- Blog des Autors zum Thema Online-Recht für Webmaster
<http://webmaster-onlinerecht.de>
- Blog des Autors zum Verbraucherrecht online
<http://verbraucherrechte-online.de>
- Weitergehende Informationen zum Thema E-Commerce
<http://rechtssicher.info>

testen und am häufigsten genutzten Online-Dienste, auch zur Entfernung der betreffenden Inhalte aufgefordert werden. Es sollte also zumindest ein Löschantrag gegenüber Google gestellt werden.

Rechtsmissbrauch

Es kommt vor, dass eine Abmahnung aus sachfremden Motiven verschickt wird. Dies ist etwa dann der Fall, wenn mit Hilfe der Abmahnung lediglich Anwaltsgebühren generiert werden sollen, um den Abgemahnten dadurch zu schädigen. Es gibt diverse Kriterien, anhand derer das Vorliegen eines rechtsmissbräuchlichen Verhaltens bewertet wird:

- fehlende oder fragwürdige Abmahnberechtigung,
- überhöhter Streitwert,
- Einfordern von nicht angefallenen beziehungsweise nicht erstattungsfähigen anwaltlichen Gebühren,
- gleicher Tag für Frist zur Abgabe der Unterlassungserklärung und für Erstattung der Abmahnkosten,
- fehlende Original-Vollmacht,
- große Anzahl gleicher oder ähnlicher Abmahnungen,
- Verwendung von Textbausteinen/ Mustertexten,
- Missverhältnis zwischen Umsätzen des Abmahnenden und seinem Kostenrisiko aufgrund der Abmahnungen,
- Betreiben eines Fake-Webshops, um ein Wettbewerbsverhältnis zum Abgemahnten zu konstruieren.

Liegen einzelne dieser Aspekte vor, bedeutet das nicht automatisch, dass ein Rechtsmissbrauch vorliegt. Je mehr Punkte zutreffen, desto eher kann jedoch davon ausgegangen werden. Vor Gericht sind Aspekte des Rechtsmissbrauchs von Amts wegen zu prüfen; bei Vorliegen eines Verdachts empfiehlt es sich allerdings, auch selbst darauf hinzuweisen. ■



Michael Rohrlisch

ist Rechtsanwalt und Fachautor aus Würselen. Seine beruflichen Schwerpunkte liegen auf dem Gebiet des Online-Rechts und des gewerblichen Rechtsschutzes.

www.ra-rohrlich.de

ARBEITSMARKT

TRENDS UND JOBS FÜR ENTWICKLER

Trends am Arbeitsmarkt

Bürokratie bremst Einstellungen

Während die Einstellung einer Fachkraft im Inland in 68 Prozent aller Fälle innerhalb von neun Monaten abgeschlossen ist und in 23 Prozent der Fälle sogar weniger als drei Monate dauert, ist der Zeitaufwand im Ausland deutlich höher (Bild 1). Bei Bewerbern aus dem EU-Ausland ist die Stelle nur in jedem zweiten Fall binnen neun Monaten besetzt. Kommt der Kandidat aus einem Nicht-EU-Land, zieht sich die Rekrutierung laut Bitkom deutlich länger hin: Nur in 17 Prozent der Fälle dauert das Verfahren weniger als neun Monate.

Bitkom-Chef Dr. Bernhard Rohleder: »Leider ist der bürokratische Aufwand im Einstellungsprozess ausländischer Fachkräfte noch so hoch, dass es oft zu extremen Verzögerungen kommt.« Das gehe von der Anerkennung von Abschlüssen bis zur Beantragung

der Arbeitsgenehmigung. Rohleder fordert: »Bürokratische Hürden bei der Einstellung von Fachkräften aus dem Ausland müssen schnellstmöglich abgebaut werden.«

Jobs für Web- und Mobile-Entwickler

Die Auswertung der Datenbank von Jobkralle.de zeigte, dass weiterhin in den Bundesländern Bayern, NRW und Baden-Württemberg die meisten Entwickler gesucht werden. Beim Vergleich der Stellenangebote für Jobs in Großstädten lag diesmal allerdings Berlin vor München. Bei den Webentwicklern entfielen 18,4 Prozent auf Berlin, München kam dagegen nur auf 17 Prozent.

Bei der Suche nach Jobs für Mobile-Entwickler entfielen sogar 19,6 Prozent der Angebote auf Berlin – in absoluten Zahlen waren das 531 Treffer, während das zweitplatzierte Hamburg auf 18,3 Prozent und München auf dem dritten Platz nur auf 17,2 Prozent kam (Bild 2).

Der Stadt-Land-Vergleich zeigte, dass 66,8 Prozent der Angebote für Webentwickler-Jobs in den 15 größten deutschen Städten ausgeschrieben wurden. Bei den Jobs für Mobile-Entwickler ist der Anteil der Angebote in Großstädten sogar noch üppiger: 73,4 Prozent.

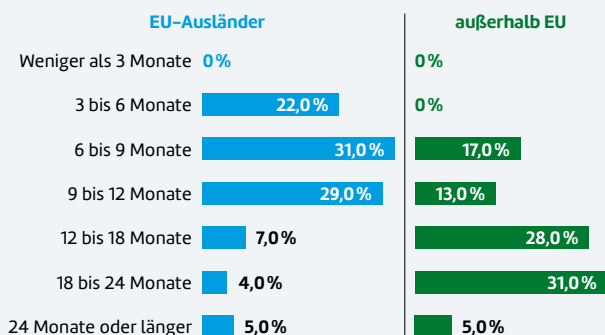
In Tabelle 1 finden Sie eine Übersicht, welche Technologie-Begriffe in Stellenangeboten besonders häufig auftauchen. Abgefragt wurden sie aus der Datenbank der Meta-Suchmaschine Jobkralle.de. Ganz oben steht der Begriff Cloud. Wer sich auf eine Stelle bewirbt, sollte sich also mit den Begrifflichkeiten rund um die Cloud auskennen. Die am häufigsten genannte Datenbank-Technologie ist MySQL, gefolgt vom Begriff HTML5. Einen großen Sprung nach vorne gemacht hat in diesem Monat das Smartphone-Betriebssystem iOS, das diesmal auf dem fünften Rang steht, in den Vormonaten dagegen nur auf Platz 7 beziehungsweise 8 zu finden war.

Tabelle 1: Technologien

Rang	Technologie	Anteil *
1	Cloud	17,6 %
2	MySQL	12,7 %
3	HTML5	10,2 %
4	SharePoint	7,8 %
5	iOS	7,4 %
6	Big Data	7,3 %
7	Android	6,7 %
8	Microsoft SQL Server	6,5 %
9	CSS3	4,9 %
10	Windows 10	3,9 %
11	AngularJS	3,7 %
12	WPF	2,8 %
13	NoSQL	2,7 %
14	Responsive Web	2,4 %
15	Azure	2,0 %
16	WCF	1,5 %

* Prozentualer Anteil der Treffer

Dauer der Fachkräfte-Rekrutierung

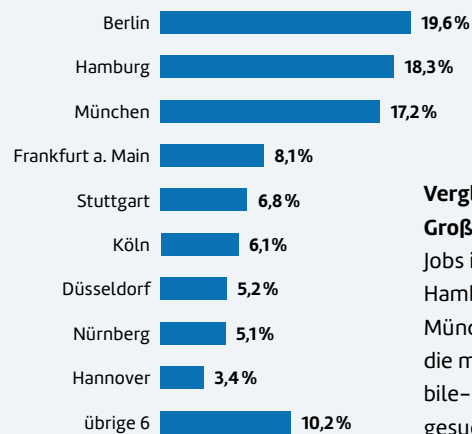


Wer Fachkräfte im Ausland rekrutieren will, braucht Geduld. Besonders lange dauert es in Nicht-EU-Ländern (Bild 1)

web & mobile developer 9/2016

Quelle: Bitkom

Mobile-Entwickler: Angebote in Großstädten



Vergleich der Großstädte: Für Jobs in Berlin, Hamburg und München werden die meisten Mobile-Entwickler gesucht (Bild 2)

web & mobile developer 9/2016

Quelle: Eigene Recherchen, Jobkralle.de

Zahl des Monats

Als Wünsche an den Vorgesetzten nennen **85 Prozent** besseren Zugang zu Informationen, selbstständig entscheiden zu dürfen und regelmäßiges Feedback. Auch flexiblere Arbeit (**71 Prozent**) und mehr Unterstützung (**60 Prozent**) stehen hoch im Kurs.

Quelle: TNT Infratest, Microsoft – Mehrfachnennungen möglich

Robert Half Technology

Neue IT-Jobs noch im Jahr 2016

Fast die Hälfte der deutschen IT-Verantwortlichen möchten noch in diesem Jahr zusätzliche Stellen schaffen. Weitere 42 Prozent planen frei werdende Positionen neu zu besetzen. Dies ergab eine aktuelle Studie im Auftrag des Personaldienstleisters Robert Half Technology. Insgesamt planen 87 Prozent der IT-Chefs, weitere Stellen zu schaffen oder freigewordene Stellen neu zu besetzen. Dies spiegelt den Optimismus der Unternehmen hinsichtlich der Wachstumsperspektiven wider: 91 Prozent der IT-Verantwortlichen sind sehr oder einigermaßen zuversichtlich, dass ihr Unternehmen in den nächsten zwölf Monaten weiter wächst. Besonders neue Projekte oder Initiativen und die nationale Geschäftsexpansion treiben die Vergrößerung der IT-Teams voran (Bild 3).

www.roberthalf.de

DEKRA Arbeitsmarkt-Report

Jobs rund ums IoT

Elektronische Bauteile, die miteinander kommunizieren, sind aus Industrie, Handwerk und Endprodukten nicht mehr wegzudenken. Dies beschert Elektronikern rosige Aussichten am Arbeitsmarkt. Wie der DEKRA Arbeitsmarkt-Report 2016 berichtet, verschwimmen in den Elektroberufen die Grenzen zu den Nachbardisziplinen wie IT oder Maschinenbau etwas, weshalb Fachkräfte mit diesen Spezialisierungen auch außerhalb ihrer jeweiligen Kernbranche interessante Job-Angebote finden. In der IT haben Software-Entwickler die größte Auswahl an Stellenangeboten. Außerdem entfällt fast jede vierte Position auf IT-Fachleute. Daneben sind insbesondere IT-Analysten und -Berater gefragt (Bild 4). Für 15,6 Prozent der IT-Positionen müssen Kandidaten betriebswirtschaftliches Know-how mitbringen.

www.dekra.de

PTV

Ausgezeichnetes IT-Praktikum

PTV ist das Unternehmen, dessen Umfragewerte die größte Übereinstimmung mit den Wünschen und Anforderungen der über 16.000 befragten Studierenden erreicht hat.

»Um mit den Branchenriesen mithalten, bieten wir als mittelständisches Unternehmen Benefits, die nicht überall selbstverständlich sind, beispielsweise eine Kantine und unser hauseigenes Fitness-Studio. Darüber hinaus haben wir uns schon seit 2014 freiwillig dazu verpflichtet, die Fair-Company-Regeln zu erfüllen. Dazu zählt unter anderem, dass wir unsere Praktikanten als vollwertige Teammitglieder behandeln, Praktikumspläne erstellen und keine Absolventen, die sich auf eine feste Stelle beworben haben, mit einem Praktikum vertrösten«, so PTV-Personalchefin Doris Prokosch.

www.ptvgroup.com

IKT-Gründungen durch Frauen

Leichter Anstieg auf 13 Prozent

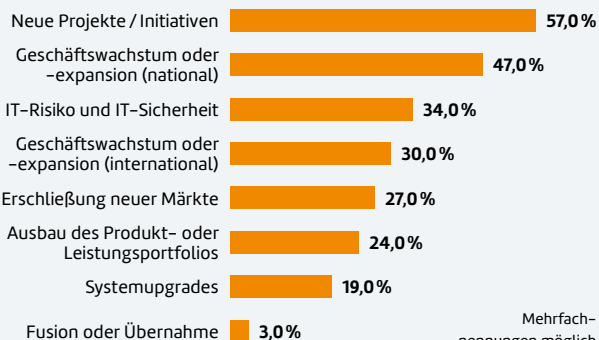
Frauen sind unter den Gründern nach wie vor in der Minderheit. Von 2014 auf 2015 ist laut Deutscher Start-up Monitor zwar ein leichter Anstieg der Gründerinnenzahl zu erkennen – von 10,7 Prozent auf 13 Prozent – doch von einem Trend lässt sich auf dieser Basis noch nicht sprechen.

Dabei schrecken Frauen keinesfalls vor der Selbstständigkeit zurück, denn im allgemeinen Gründungsgeschehen sind Frauen mit 43,3 Prozent vertreten. Noch höher sind die Beteiligungen von Frauen an Existenzgründungen durch freiberufliche Tätigkeiten.

Mit dem Gründerwettbewerb – Digitale Innovationen, einem Ideenwettbewerb des BMWi, sollen Frauen gezielt dabei unterstützt werden, ihre Vorhaben im IKT-Bereich umzusetzen.

www.gruenderwettbewerb.de

Schaffung neuer IT-Jobs

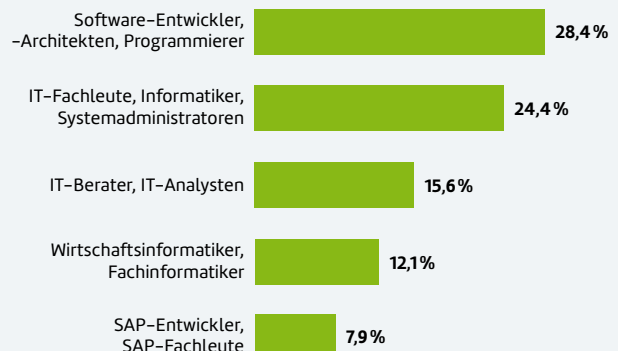


Neue Projekte und das Wachstum der Unternehmen sind die häufigsten Gründe für die Schaffung neuer IT-Jobs (Bild 3)

web & mobile developer 9/2016

Quelle: Robert Half, Arbeitsmarkstudie

Gesuchte IT-Fachkräfte



Entwickler, Programmierer und Software-Architekten sind die am häufigsten gesuchten IT-Fachkräfte (Bild 4)

web & mobile developer 9/2016

Quelle: DEKRA Akademie 2016

Stellenmarkt

dotnetpro + web & mobile Developer

○ 25.800 Exemplare Gesamtauflage

○ 25.300 Newsletter-Empfänger

○ 66.600 PI'S



○.NET ○Architektur ○HTML5/JavaScript ○iOS/Android ○

Kontakt:

Jens Schmidtman, Klaus Ahlering • Tel. 089/74117-125 • sales@nmg.de

Anbieterverzeichnis

für Deutschland, Schweiz und Österreich.

Consulting / Dienstleister



ANEXIA Internetdienstleistungs GmbH

Feldkirchner Straße 140
9020 Klagenfurt / AUSTRIA
T +43-50-556
F +43-50-556-500
info@anexia-it.com

ANEXIA wurde im Juni 2006 von Alexander Windbichler als klassischer Internet Service Provider gegründet. In den letzten Jahren hat sich ANEXIA zu einem stabilen, erfolgreichen und international tätigen Unternehmen entwickelt, das namhafte Kunden rund um den Globus mit Standorten in Wien, Klagenfurt, München, Köln und New York City betreut. ANEXIA bietet ihren Kunden hochwertige und individuelle Lösungen im Bereich Web- und Managed Hosting, sowie Individualsoftware und App Entwicklung.



prodot GmbH

Schifferstraße 196
47059 Duisburg
T: 0203 - 346945 - 0
F: 0203 - 346945 - 20
info@prodot.de
https://prodot.de

prodot – People. Passion. Performance.

Intelligente Software für internationale Konzerne und mittelständische Unternehmen: prodot stärkt Kunden im weltweiten Wettbewerb – mit effizienten, stabilen und kostensenkenden Lösungen. Durch das Zusammenspiel aus Know-how, Kreativität und Qualitätsmanagement leisten wir einen Beitrag zum langfristigen Erfolg unserer Auftraggeber. prodot bringt hierzu die richtigen Menschen zusammen. Seit über 15 Jahren vertrauen uns deshalb Marktführer wie Aldi Süd, Microsoft und Siemens. Sprechen Sie mich an – als Kunde, Partner oder Kollege. Pascal Kremmers.

eCommerce / Payment



Payone GmbH & Co. KG

Fraunhoferstraße 2-4
24118 Kiel
T: +49 431 25968-400
F: +49 431 25968-1400
sales@payone.de
www.payone.de

PAYONE ist einer der führenden Payment Service Provider und bietet modulare Lösungen zur ganzheitlichen Abwicklung aller Zahlungsprozesse im E-Commerce. Das Leistungsspektrum umfasst die Zahlungsabwicklung von allen relevanten Zahlarten mit integriertem Risikomanagement zur Minimierung von Zahlungsausfällen und Betrug. Standardisierte Schnittstellen und SDKs erlauben eine einfache Integration in bestehende IT- und mobile Systemumgebungen. Über Extensions können auch E-Commerce-Systeme wie Magento, OXID eSales, Demandware, Shopware, plentymarkets und viele weitere unkompliziert angebunden werden.

Web- / Mobile-Entwicklung & Content Management



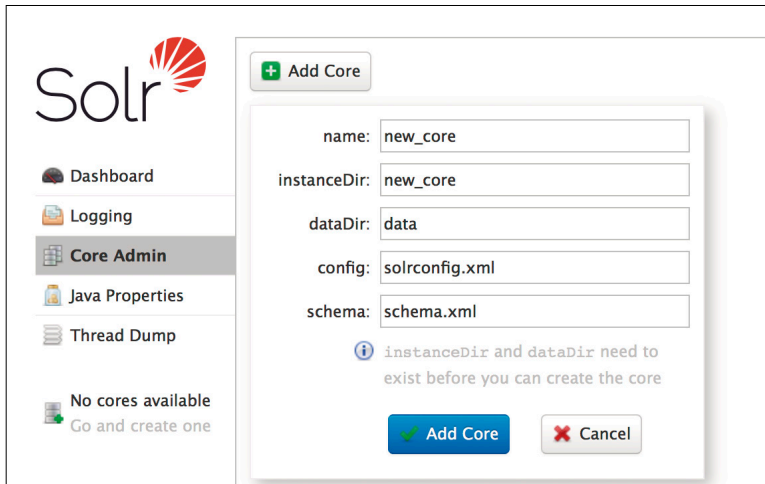
digitalmobil GmbH & Co. KG

Bayerstraße 16a, 80335 München, T: +49 (0) 89 7 41 17 760, info@digitalmobil.com, www.digitalmobil.com

In allen Fragen rund um das Dienstleisterverzeichnis berät Sie Frau Roschke gerne persönlich!
Juliane Roschke ■ 089 / 7 41 17 - 283 ■ juliane.roschke@nmg.de

Die Ausgabe 10/2016 erscheint am 8. September 2016

Solr mit PHP



Solr, eine der beliebtesten Enterprise-Suchen, gepaart mit PHP, der beliebtesten Programmiersprache im Web, stellt eine harmonische Kombination dar. Solr ist eine in Java geschriebene Open-Source-Enterprise-Search-Plattform, die das Apache-Lucene-Projekt erweitert. Hauptmerkmale sind Volltextsuche, Treffer-Highlighting, facettierte Suche, Indexing in Echtzeit, dynamisches Clustering, Datenbank-Integration, No-SQL-Unterstützung und Rich-Document-Handling. Solr unterstützt verteilte Suche sowie Index-Replikation und ist für Hochskalierbarkeit und Fehlertoleranz konzipiert worden. Weltweit ist Solr nach Elasticsearch die beliebteste Suchmaschine. Die externe Konfiguration von Solr erlaubt die Anpassung an eine Vielzahl von Applikationen.

iOS SpriteKit Scene Editor

SpriteKit ist die Bibliothek, um Spiele für iOS, tvOS, watchOS und macOS zu schreiben. Apple unterstützt den Entwickler mit vielen nützlichen Funktionen und Klassen, die ihm viel Programmier-Arbeit abnehmen. Apple hat die SpriteKit-Bibliothek seit der ersten Vorstellung stark weiterentwickelt. Aber nicht nur die Bibliothek wurde erweitert, auch die Unterstützung durch Xcode bei der Entwicklung von Spielen wurde vorangetrieben.

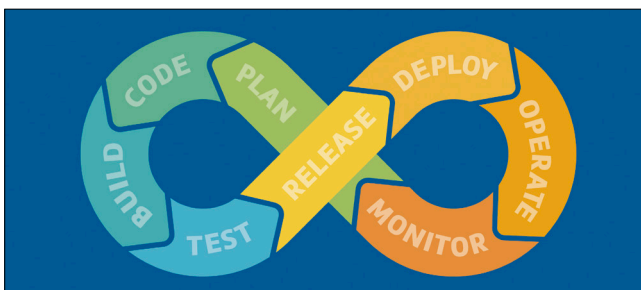
Command Line API

Browser-Tools wie die Chrome Developer Tools, Firebug oder die Entwickler-Tools von Safari gehören zu den Standardwerkzeugen eines jeden Webentwicklers. Sei es, um den DOM-Baum einer Webseite zu inspizieren, Ausgaben auf der Konsole zu erzeugen oder um Breakpoints für das Debuggen zu definieren. Das Command Line API stellt unterschiedliche Funktionalitäten diverser Browser-Entwicklertools per JavaScript zur Verfügung.

Websockets in HTML5/JavaScript

Bei einer reinen HTTP-Verbindung wird bei jeder Aktion des Servers eine Anfrage des Clients benötigt. Beim WebSocket-Protokoll ist es ausreichend, wenn der Client die Verbindung öffnet. Das WebSocket-Protokoll ermöglicht bidirektionale Verbindungen zwischen einem Server und einer Webanwendung. WebSockets sind Teil des HTML5-Standards. Ein Artikel erklärt die Funktionalität und das grundlegende Vorgehen mit Websockets.

dotnetpro



Ausgabe 9/2016 ab 18. August 2016 am Kiosk

DevOps fügen dem Application Life Circle aus Planung, Implementierung, Bauen und Testen noch eine Schleife aus Verteilung, Betrieb und Überwachung hinzu. Sie machen aus dem Kreis ein Unendlich-Zeichen. Wie DevOps und Entwickler am besten zusammenarbeiten. www.dotnetpro.de

Unsere digitalen Angebote



Wöchentlicher Newsletter
webundmobile.de/newsletter



Shop
shop.webundmobile.de



YouTube
youtube.com/user/developermedia



Facebook
facebook.com/webundmobile



Google +
gplus.to/webundmobile



Twitter
twitter.com/webundmobile

VDI – Vereinfachen Sie die Migration zu Windows 10 und darüber hinaus



Machen Sie sich schnell und einfach bereit für Windows-10-Endgeräte, -Anwendungen und -Desktops mit Desktop-Virtualisierung

GRATIS
Whitepaper

Lesen Sie im kostenlosen Whitepaper von **CITRIX®**

- Optimierte Betriebssystemmigration durch Virtualisierung
- Der richtige Umstieg auf Windows 10
- Integration unterschiedlicher Endgeräte (inkl. BYOD)
- Migration der Anwendungen
- Standardisierung der Desktop-Images

Jetzt kostenlos downloaden:
<http://digital.internetworld.de/citrix-wp-vdi-wump/>

Ein Service von:

 **INTERNET WORLD Business**

Ihr Whitepaper bei uns?
Anfragen an sales@nmg.de
oder Tel. 089 / 7 41 17 – 124.

Ihr Partner für mehr Online-Wachstum

Wir planen, entwickeln und steuern
Websites, Apps und Kampagnen.

Unsere Ziele sind Ihre Ziele:

- ⊕ **Mehr Sichtbarkeit**
- ⊕ **Mehr Traffic**
- ⊕ **Mehr Leads**
- ⊕ **Mehr Conversions**

.....

➔ **Mehr Kunden**

Besuchen Sie uns unter
www.digitalmobil.com

