02-2016 | Sommer | www.ijug.eu

Java aktuell

Praxis. Wissen. Networking. Das Magazin für Entwickler

Aus der Community – für die Community

Java aktuell

D: 4,90 EUR A: 5,60 EUR CH: 9,80 CHF Benelux: 5,80 EUR ISSN 2191-6977

Java ist Programmiersprache des Jahres

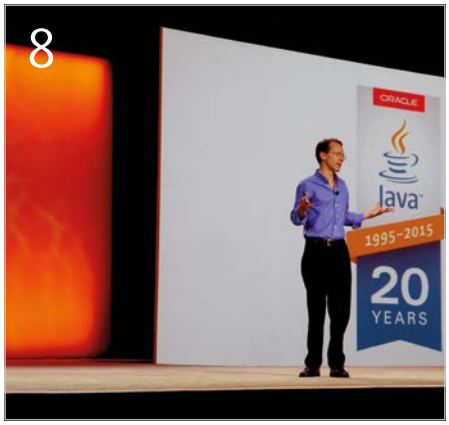
Sicherheit
- Social Login
- Single Sign-on

Richtig testen
Statische Code-Analyse

Richtig entwickeln
Lösung mit JavaFX und JVx

4 191978 304903 02

iJUG Verbund



Neues von der JavaOne

Die gängigsten Open-Source-Tools zur Code-Analyse im Praxis-Einsatz

Reaktive Anwendungen mit asynchronen, Event-getriebenen Architekturen gewinnen stark an Bedeutung

Leuchtfeuer in Innenräumen: Micro-location-based Services mit Beacons

Constantin Mathe und Bernd Müller, Ostfalia

Mithilfe sogenannter „Beacons" verschmilzt das Internet mit der direkten Umgebung des Benutzers. Die Beacon-Technologie wird in den nächsten Jahren grundlegend neue Möglichkeiten entstehen lassen, um mit dem Internet und der direkten Umwelt zu interagieren. Die Konzerne Apple und Google prognostizieren einen Zuwachs von Beacons in Millionenhöhe für die nahe Zukunft. Doch was verbirgt sich hinter Beacons und den angepriesenen Vorteilen für den Anwender?

Stellen Sie sich vor, Sie stehen in einem Museum und erhalten automatisch die passenden Informationen über das entsprechende Exponat oder bekommen beim Aufenthalt in einem Bahnhof Informationen über mögliche Reisemöglichkeiten direkt auf Ihr Smartphone. All dies und viel mehr ist bereits heute mit Beacons möglich. Dieser Artikel führt die zugrunde liegende Technologie ein und zeigt exemplarisch, wie sie verwendet werden kann.

Location Based Services/ Micro-Location

Welche Möglichkeiten ergeben sich, wenn wir in der Lage wären, mithilfe unseres Smartphones direkt mit unserer näheren Umgebung zu interagieren? Es ist längst kein Problem mehr, über GPS und andere Sensoren den ungefähren Standort des Benutzers zu ermitteln. Mit aktuellen Smartphones ist es etwa ohne Probleme möglich zu erkennen, ob man sich in einem Supermarkt, einem Kino oder einer Hochschule befindet. Doch wenn es darum geht zu ermitteln, ob man vor der Obsttheke oder der Fleischtheke, im Kinosaal A oder B oder im Hörsaal für Mathematik oder Informatik steht, hört es mit der Genauigkeit von GPS und Mobilfunkzellen auf. Hier beginnt „Micro-Location" interessant zu werden.

Um kontextbezogene Dienste in diesem Genauigkeitsbereich anzubieten, muss das Smartphone die Möglichkeit besitzen, den Benutzer auf wenige Zentimeter genau zu

lokalisieren. Location-based Services werden zu Micro-Location-based Services. Ziel ist es, dem Benutzer anhand seiner exakten Position die Interaktion mit seiner unmittelbaren Umgebung zu ermöglichen. Diese Contextual Services hängen damit mit der aktuellen Vision des Physical Web zusammen und sind ein zentraler Baustein zu dessen Verwirklichung.

Zur Veranschaulichung der Möglichkeiten vergleichen wir die aktuell gängigen und verwendeten Technologien. Mittels GPS sind wir in der Lage, bis auf etwa zehn Meter genau eine Position zu bestimmen. Diese Genauigkeit ermöglicht Dienste, die für die oben genannten Beispiele der Lokalisierung von Supermarkt, Kino oder Hochschule sinnvoll sind.

Eine GPS-Ortung funktioniert in vielen Fällen mehr schlecht als recht. Zum Beispiel sind in größeren Gebäuden mit guter Isolation kein GPS-Empfang und damit auch keine entsprechenden Dienste-Angebote möglich. Eine andere Möglichkeit wäre WLAN. Damit lässt sich zwar eine Lokalisierung in Gebäuden erreichen, allerdings ist diese Genauigkeit von rund fünfzig Metern für entsprechende Dienste nicht ausreichend.
Genau hier füllt Bluetooth als Basis-Technologie die Lücke zur Micro-Location. Mittels Bluetooth ist eine Ortsbestimmung bis auf wenige Zentimeter genau möglich. Wie das genau funktioniert, werden wir gleich sehen. Zur Verdeutlichung zeigt *Abbildung 1* noch einmal die Lokalisierungsgenauigkeiten der angesprochenen Technologien im Überblick.

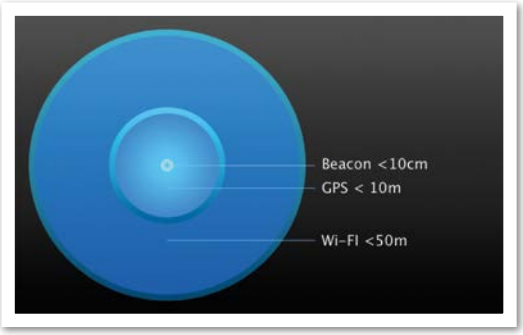


Abbildung 1: Lokalisierung entsprechend den Technologien

Wer Beacons erfunden hat

Das englische Wort „Beacon" bedeutet auf Deutsch „Leuchtfeuer, Leuchtturm oder Blinklicht" und beschreibt einen von Apple im Jahr 2013 eingeführten Standard für die Navigation in geschlossenen Räumen. Beacons senden ein auf Bluetooth Low Energy (BLE) basierendes Signal aus, das von anderen BLE-fähigen Geräten empfangen und von entsprechenden Anwendungen verarbeitet werden kann. Das übermittelte Signal enthält die Identifikation des Beacon sowie eine Empfangsstärke (Received Signal Strength Indicator, RSSI). Ein Beacon selbst ist nicht in der Lage, Inhalte auszuliefern oder Benutzer zu lokalisieren und zu verfolgen. Es liegt also an uns Entwicklern, mithilfe von Anwendungen das gewünschte Verhalten zu realisieren.

Da es sich bei Bluetooth um ein grundlegend offenes Protokoll handelt, wird BLE bei iOS ab Version 7 und bei Android ab der Version 4.3 unterstützt. Alle aktuell verfügbaren Mobilgeräte mit Bluetooth unterstützen damit den Standard von Haus aus. Aktuelle Computer sind ebenfalls BLE-fähig. Die neueren Versionen von Windows, OS X und Linux haben die entsprechenden Treiber eingebaut.

Die Anwendungsszenarien von Beacons sind relativ breit gestreut: im Restaurant, Theater, Hotel, Schwimmbad, Arztpraxis, Krankenhaus, Universität, Supermarkt oder Kino. Beacons lässt sich nahezu überall für die Verbreitung unterschiedlichster Informationen einsetzen.

Da im Gegensatz zu normalen Bluetooth-Geräten bei Beacons keine Kopplung zwischen den an der Kommunikation beteiligten Geräten stattfindet, kommt bei Beacons ein anderer Bluetooth-Paket-Typ zum Einsatz. Es handelt sich um ein Advertising-Bluetooth-Package, also ein Broadcast-Paket. Um die Größen-Anforderungen eines einzelnen Pakets zu minimieren, wurde ein hochkomprimiertes Format definiert. *Abbildung 2* zeigt den schematischen Aufbau, den wir im Folgenden erläutern.

iBeacon (Apple Standard)

Der iBeacon-Standard *[2]* sieht im Wesentlichen drei unterschiedliche IDs für ein Beacon vor. Hierzu gehört eine Identifikation des Beacon oder auch Proximity ID in Form einer UUID. Dahinter verbirgt sich eine 16-Byte-Zahl, die hexadezimal notiert und in fünf Gruppen unterteilt wird. Eine solche UUID dient als Hersteller-Kennung und hilft somit, unterschiedliche Hersteller zu unterscheiden. Der Beacon-Hersteller Estimote verwendet beispielsweise die UUID „B9407F30-F5F8-466E-AFF9-25556B57FE6D".

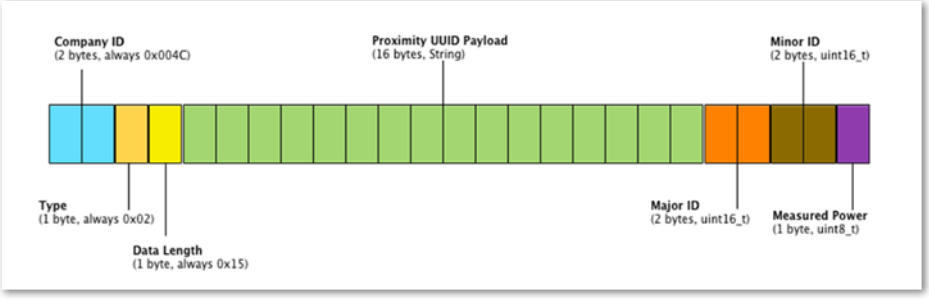


Abbildung 2: Der schematische Aufbau

Zusätzlich besitzt jedes Beacon eine Major- und eine Minor-ID. Diese werden verwendet, um die Beacons bestimmten Zonen zuordnen zu können und somit einzelne Beacons zu identifizieren. Laut offizieller Darstellung von Apple sollte die Major-ID zum Beispiel dazu verwendet werden, Beacons einem bestimmten Gebäude zuzuordnen. Die Minor-ID kommt dann zum Einsatz, um den Raum oder die entsprechende Etage eindeutig zu definieren. Die eigentliche Verwendungsart steht aber jeder Anwendung frei.

Weiterhin wird in jedem Advertising-Paket die Signalstärke mit ausgeliefert, sodass auf dieser Basis die Entfernung zum Empfänger berechnet werden kann. Man darf aber nicht vergessen, dass Beacons auf einem 2,4-Ghz-Kanal senden und somit relativ stark durch andere Funksignale gestört werden können. Insbesondere räumliche Hindernisse wie Möbel oder Personen beeinflussen das Signal ebenfalls.

Im Netz lassen sich zahlreiche Tests finden, in denen die Genauigkeit von Beacons unterschiedlicher Hersteller untersucht wird. Eins lässt sich mit Sicherheit sagen: Die Genauigkeit von Beacons hängt stark vom Hersteller und der Bauweise sowie der Konfiguration des Beacon ab. Parameter, wie die Sendestärke (Transmit Power) selbst und die Wiederholungsfrequenz des Signals (Advertising Interval), haben eine große Auswirkung auf die Qualität des Signals. Beide Parameter haben auch großen Einfluss auf den Stromverbrauch und damit die Batterie-Lebensdauer. Das in iOS integrierte Framework von Apple bietet aufgrund der schlecht vorhersagbaren Genauigkeit nur drei Abstufungen von Entfernung an:

- Immediate (0-20 cm)
- Near (20 cm – 2 m)
- Far (2 – 7 m)

Anhand dessen ist schon ersichtlich, dass eine genaue Angabe der Distanz zwischen Beacon und Smartphone nicht immer angegeben werden kann. Wie kommen diese gravierenden Unterschiede zustande? Es gibt einige Unterschiede zwischen den SDKs der einzelnen Beacon-Hersteller:

- Drop-out-Rate
- Implementierung der Empfangsstärke (RSSI)
- Konsistenz von Beacons desselben Herstellers
- Ausrichtung der Antenne im Beacon
- Störsignale unter den Beacons

Die folgende Gleichung beschreibt, wie die Distanzmessung zwischen Beacon und Empfänger funktioniert „P(d)[dBM] = P(d0)[dBM] – 10n log(d/d0) – Z", wobei die einzelnen Parameter Folgendes bedeuten:

- D = erwartete Distanz
- d0 = ein Maß für eine Distanz für eine bekannte Signalstärke
- n = Parameter für den Signalstärkenverlust über steigende Distanz
- Z = Umwelt-Einflüsse

Es ist relativ schwierig, einen guten Algorithmus für die Distanz-Messung zu finden, der auch noch für einen Großteil der Geräte und verbauten Antennen gut funktioniert. Es hat sich aber gezeigt, dass die folgende Java-Methode eine sehr gute Annäherung an die wirkliche Distanz ermöglicht *(siehe Listing 1)*.

Der Code stammt von *[1]* und wurde durch Tests parametrisiert, die die Autoren durchführten. Die Konstanten „0.89976", „7.7095" und „0.111" wurden durch Ausgleichsrechnung, basierend auf den Testergebnissen, definiert. Sie sind also durch die Praxis validiert und keine Natur-Konstanten.

```
/* Calculate Distance of iBeacons */
protected static double calculateAccuracy(int txPower, double rssi) {
   if (rssi == 0) {
       return -1.0; // if we cannot determine accuracy, return -1.
   }

   double ratio = rssi*1.0/txPower;
   if (ratio < 1.0) {
       return Math.pow(ratio,10);
   }
   else {
       double accuracy = (0.89976)*Math.pow(ratio,7.7095) + 0.111;
       return accuracy;
   }
}
```

Listing 1

Eddystone

Im Juni dieses Jahres kam von Google relativ überraschend ein eigener Beacon-Standard namens „Eddystone" auf den Markt. Hierbei handelt es sich um eine direkte Alternative und damit Konkurrenz zum iBeacon-Standard von Apple. Im Gegensatz zu Apple versucht Google allerdings, einen Standard zu schaffen, der sowohl mit iOS als auch mit Android kompatibel ist. Ein weiterer Vorteil von Eddystone ist, dass er quelloffen auf GitHub unter einer Apache-2.0-Open-Source-Lizenz zur Verfügung steht *[3]*, während Apples Implementierung nicht öffentlich ist.

Während Beacons auf Basis des iBeacon-Standards eine zusätzliche App benötigen, geht Eddystone einen Schritt weiter und kann Informationen auch ohne zusätzliche App versenden. Dies ist möglich, da Eddystone im Gegensatz zu iBeacons nicht nur die üblichen UUIDs versenden, sondern im direkten Vergleich auch Internet-Adressen in Form von URLs ausstrahlen kann. Dies ermöglicht es, direkt Werbe-Inhalte an Anwender zu senden, ohne eine App dazwischenschalten zu müssen. Es entfällt also die Web-Komponente in Form eines zentralen Registers für UUIDs. Dieser Schritt ebnet den Weg zum Physical Web *[4]*. Es bleibt aber zu erwähnen, dass selbst URL-Frames nicht ohne die Chrome-App unter iOS lesbar sind. Es ist also nur unter Android möglich, ohne App Informationen zu erhalten, da hier Google bereits die nötigen Automatismen in das Betriebssystem eingebaut hat.

Eddystone Frames

Wie schon erwähnt, kann Eddystone auch ohne Umwege direkt Internet-Adressen ohne UUID ausstrahlen. Realisiert wird dies durch unterschiedliche Frame-Typen. Ein Eddystone-Beacon kann zwischen drei unterschiedlichen Frames wählen, die gesendet werden sollen:

- UID-Frame
- URL-Frame
- TLM-Frame

Der UID-Frame kommt dem des iBeacon am nächsten. Hierbei wird eine ID ausgesandt, die sich aus zwei weiteren IDs zusammensetzt. Die Namespace-ID entspricht hierbei einer Proximity-ID, während die Instance-ID dazu verwendet wird, zwischen einzelnen Beacons zu unterscheiden.

Der URL-Frame ermöglicht es, eine Internet-Adresse zu kodieren und diese anschließend direkt über das Bluetooth-Advertising-Paket zu verschicken. Android ist von Haus aus in der Lage, diese URLs aus den Paketen zu dekodieren und dann als Physical-Web-Objekte gruppiert darzustellen.

Beim TLM-Frame handelt es sich um einen speziellen Frame, der für die Überwachung der Beacons zum Einsatz kommt. Dieser Frame enthält verschiedene Hardware-nahe Informationen, etwa über Batteriestand, Temperatur, Advertising-PDU-Zähler und Uptime in Sekunden. Eddystone Beacons senden also nicht nur eine UID oder URL, sondern in konfigurierbaren Sekunden-Intervallen zusätzlich einen TLM-Frame mit Hardware-Informationen. Dieses Feature ermöglicht es erst, große Installationen von Beacons zu überwachen, um beispielsweise herauszufinden, wann die Batterien ausgewechselt werden müssen. Um schnell mit Eddystone Beacons starten zu können, sind auf der GitHub-Seite *[3]* mehrere Code-Beispiele für Android und iOS verfügbar.

Aktuell verfügbare Hardware (Beacons)

Mittlerweile gibt es eine große Anzahl von Beacon-Herstellern. Egal ob wetterfest, wasserdicht, mit Batteriebetrieb oder fester Stromversorgung mit Micro-USB – fast alles ist verfügbar. Doch was sollte man vor dem Kauf beachten? Viele Hersteller bringen ein eigenes Framework mit, um die Beacons direkt mit möglichst wenig Aufwand in bestehende Apps einzubinden. Doch nicht jedes Framework ist kompatibel mit iOS und Android. Außerdem muss man als Entwickler die Entscheidung treffen, ob man von den hauseigenen Cloud-Services der Hersteller abhängig sein möchte oder nicht.
Hinzu kommt, dass man darauf achten sollte, dass die Beacons über Firmware-Updates erweiterbar und somit Update-fähig sind. Da Googles Eddystone noch nicht finalisiert wurde, kann man davon ausgehen, dass es noch einige Änderungen und Er-

Abbildung 3: Der offene Estimote-Beacon, Quelle: http://estimote.com

weiterungen am Protokoll geben wird. Hier macht es sich bezahlt, wenn die Beacon-Hardware per Firmware- beziehungsweise Software-Update aktualisiert werden kann. Die Firma Estimote *[6]* hat Beacons und Nearables im Sortiment, die mit einem eigenen quelloffenen und auf GitHub gehosteten SDK leicht in vorhandene Projekte einbindbar sind. Außerdem ist das Framework Android- und iOS-kompatibel. Ein Developer-Kit, bestehend aus drei Beacons, ist bereits ab 99 Dollar im Angebot. Bei Testkäufen der Autoren wurden die Beacons innerhalb weniger Werktage geliefert. Estimote bietet auch eine eigene kostenlose App für iOS und Android an, um die Einstellungen der Beacons zu konfigurieren, also beispielsweise die Parameter. *Abbildung 3* zeigt ein Estimote-Beacon in geöffnetem Zustand.

Verwendung mit Java

Estimote stellt unter *[8]* ein SDK für Android zur Verfügung. Das API hinterlässt einen guten Eindruck und basiert hauptsächlich auf der Registrierung von Event-Listenern, die für Beacon-Sende- beziehungsweise Empfangsbereiche und Empfangsstärken parametrisiert werden können. Die Autoren können hier leider noch keine Beispiele präsentieren, hoffen aber, dies in einer der nächsten Ausgaben der Java aktuell nachholen zu können.

Was bringt die Zukunft?

Was die Zukunft angeht, rechnet man damit, dass bis zum Jahr 2018 allein in Deutschland die Anzahl an Beacons auf sechzig bis dreihundert Millionen ansteigen wird *[9]*. Nachdem Google in den Beacon-Markt eingestiegen ist, kann davon ausgegangen werden, dass das Unternehmen in den kommenden Monaten eventuell eigene Beacon-Hardware, basierend auf dem hauseigenen Framework, auf den Markt bringen wird.

Es bleibt jedoch das Problem, dass keines der beiden Protokolle, weder iBeacon noch Eddystone, plattformübergreifend ohne eine spezielle App funktioniert. Eddystone ist zwar iOS-kompatibel, benötigt aber aktuell noch die Chrome-App, die auf dem iOS Gerät installiert werden muss. Umgekehrt gilt dasselbe für Android mit iBeacons. Diese werden nur über Apps von Dritten gefunden. Es bleibt also die Hoffnung, dass sich beide Firmen auf einen Standard einigen und diesen dann jeweils in ihrem Betriebssystem fest verankern. Eine plattformübergreifende Nutzung von Beacons und deren Informationen würde das Physical Web dramatisch nach vorne bringen.

Fazit

Der durch Apple veröffentlichte iBeacon-Standard ermöglicht Micro-location-based Services beliebiger Couleur. Es zeichnet sich erst langsam ab, welch breites Spektrum an Anwendungen damit realisierbar ist. Nachdem Google auf den stark beschleunigenden Zug aufgesprungen ist, besteht kein Zweifel daran, dass Beacons zu einem großen Pfeiler in dem gerade gehypten Internet of Things werden. Mit Eddystone steht Java-Entwicklern eine Open-Source-Implementierung des Protokolls inklusive SDK zur Verfügung, um mit wenig Aufwand in diese interessante Welt neuartiger Anwendungen einsteigen zu können.

Referenzen

[1] *http://stackoverflow.com/questions/20416218/understanding-ibeacon-distancing/204340199-20434019*
[2] *https://developer.apple.com/ibeacon/*
[3] *https://github.com/google/eddystone/*
[4] https://google.github.io/physical-web/
[5] *https://developers.google.com/beacons/*
[6] *http://estimote.com*
[7] *http://beaconinside.com*
[8] *https://github.com/Estimote/Android-SDK*
[9] *https://iotcon.de/2015/sessions/future-beacons*

Constantin Mathe
con.mathe@ostfalia.de

Constantin Mathe studiert Informatik an der Ostfalia (Hochschule Braunschweig/Wolfenbüttel) und schreibt gerade seine Masterarbeit über das Thema „Micro-location mit Beacons". Während seines Studiums hat er mehrere Jahre als Java-Software-Entwickler bei der Symtavision GmbH gearbeitet.

Bernd Müller
bernd.mueller@ostfalia.de

Bernd Müller ist Professor für Software-Technik an der Ostfalia (Hochschule Braunschweig/Wolfenbüttel). Er ist Autor mehrere Java-EE-Bücher, Sprecher auf nationalen und internationalen Konferenzen und engagiert sich in der JUG Ostfalen sowie im iJUG.