

Barrierefreiheitsdatenbank

Christian Schramm

April 8, 2014

Version: 0.0.1



BERUFSAKADEMIE SACHSEN Staatliche Studienakademie
Dresden

Belegarbeit Webprogrammierung

Barrierefreiheitsdatenbank

Christian Schramm

Gutachter

Maksim Gudow

BERUFSAKADEMIE SACHSEN Staatliche Studienakade-
mie Dresden

April 8, 2014

Christian Schramm

Barrierefreiheitsdatenbank

Belegarbeit Webprogrammierung, April 8, 2014

Gutachter: Maksim Gudow

BERUFSAKADEMIE SACHSEN Staatliche Studienakademie Dresden

Hans-Grundig-Straße 25

01307 Dresden

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Zielstellung und Abgrenzung der Arbeit	2
1.3	Untersuchungsgegenstand	2
1.3.1	Forschungsfrage	2
1.3.2	Hypothesen	2
2	Theorie/Begriffsbestimmung	4
2.1	Native Apps	4
2.1.1	Vorteile	4
2.1.2	Nachteile	5
2.2	Webapps	5
2.2.1	Vorteile	5
2.2.2	Nachteile	6
2.3	Hybridapps	6
2.3.1	Vorteile	6
2.3.2	Nachteile	7
2.4	Wahl der Appvariante	7
3	Synchronisation	11
3.1	Datenbanksynchronisation am Bsp. WebSqlSync	11
3.2	Dateisynchronisation am Bsp. ownCloud	13
4	Fazit	16
4.1	System Section 1	16
4.2	System Section 2	16
4.3	Ausblick	16

Einleitung

“ *There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. The first method is far more difficult.*

— C.A.R. Hoare

(British computer scientist, winner of the 1980 Turing Award)

Das Thema Barrierefreiheit betrifft einen jährlich zunehmenden größer werdenden Teil der Bevölkerung. Im Jahr 2011 gab es mit knapp 7,3 Millionen schwerbehinderten Menschen rund 2,6% mehr als noch im Jahr 2009.[@Sta11] Dabei leiden zweidrittel der schwerbehinderten Menschen unter körperlichen Behinderungen, welche je nach Grad und Art der Behinderung zu sehr starken Einschränkungen im Alltag führen können.

Im Bereich des öffentlichen Lebens wird sehr viel dafür getan, um z.B. die Zugänge zu öffentlichen Einrichtungen wie Museen, Ämtern oder öffentlichen Verkehrsmitteln zu erleichtern. In der Verordnung DIN 18040 Barrierefreies Bauen sind grundlegende Anforderungen an öffentlich zugängliche Gebäude beschrieben. Die Einführung dieser Verordnung bzw. der einzelnen Punkte in die Technischen Baubestimmungen obliegt jedoch den einzelnen Bundesländern.[@Hyp10]

Um erfassen zu können welche Einrichtungen die behindertengerechten Anforderungen erfüllen, wurde von der Thüringer Tourismus GmbH in Zusammenarbeit mit dem Dresdner Unternehmen webit! Gesellschaft für neue Medien mbH das Konzept der Barrierefreiheitsdatenbank entwickelt. Sie ermöglicht es alle Daten über eine Einrichtung zu erfassen. Dazu zählen z.B. Zugänge, Treppen, Liftanlagen und vieles mehr. Alle Daten die von Mitarbeitern des Unternehmens erfasst werden, kommen in einer Datenbank zusammen und sollen dabei helfen behinderten Menschen bei der Planung ihres Urlaubs oder von Ausflügen zu unterstützen.

1.1 Motivation

Die Überlegung zu dieser Arbeit kam durch die Erfassung der ersten öffentlich begehbaren Einrichtungen. Um zum derzeitigen Zeitpunkt Daten erfassen zu können, wird eine Internetverbindung benötigt. Diese kann aber bei vielen Einrichtungen, wie z.B. Kellergewölben, Museen oder Schlössern nicht gewährleistet werden. Dadurch entstand der Wunsch die

Daten unabhängig von einer Internetverbindung einpflegen zu können. Die aufgenommenen Bilder und Informationen zu den baulichen Gegebenheiten sollen dann mit den vorhandenen Daten auf dem Server abgeglichen werden und gegebenenfalls auf dem Client, als auch auf dem Server ergänzt werden.

1.2 Zielstellung und Abgrenzung der Arbeit

Smartphones und Tablets gehören mittlerweile zum Standard der mobilen Kommunikation. Mit diesen kleinen Alleskönnern kann man Fotos aufnehmen, im Internet surfen, Dokumente erfassen und vieles mehr, was noch vor ein paar Jahren nur von Desktoprechnern denkbar gewesen wäre. Das macht die heutige Datenerfassung sehr viel flexibler als noch vor ein paar Jahren.

Ziel der Arbeit ist es eine Möglichkeit zu finden, Daten die mit mobilen Endgeräten erfasst werden mit den Daten auf einem Server zu synchronisieren. Dabei soll sowohl die Datenbank betrachtet werden, als auch Bilddateien. Als Ausgangspunkt dienen die Vor- und Nachteile die es bei nativen Apps, Web Apps und einer Hybrid App Variante gibt und wie sich das auf eine Synchronisation der Daten auswirkt.

In dieser Arbeit wird keine native App und auch keine Webapp programmiert, sondern nur Überlegungen zur Umsetzung einer erfolgreichen Synchronisation getroffen, die dann eine Entscheidung über die Wahl der mobilen Variante der Barrierefreiheitsdatenbank beeinflussen könnte.

1.3 Untersuchungsgegenstand

1.3.1 Forschungsfrage

Nach Betrachtung der Zielstellung ergab sich eine Forschungsfrage, die am Ende der Arbeit beantwortet werden soll.

Gibt es eine Möglichkeit alle Daten die mit mobilen Endgeräten offline erfasst werden, bei bestehender Internetverbindung mit dem Server abzugleichen?

1.3.2 Hypothesen

Daraus ergaben sich folgende Hypothesen, welche sich bei der Untersuchung der Methoden als richtig oder falsch herausstellten.

1. Hypothese

Es lassen sich alle mit einem mobilen Endgerät erfassten Daten uneingeschränkt per Internet mit dem Server synchronisieren.

2. Hypothese

Die Entwicklung einer nativen App ist die beste Lösung für die Nutzung der Hardwarefunktionen und des Speicherbedarfs der Kombination aus Datenbank und Bildern.

3. Hypothese

Die Umsetzung einer Web App bietet sich aufgrund der bestehenden Website an. Dadurch verringern sich Aufwand und Kosten, ohne Einschränkung der Funktionen.

Theorie/Begriffsbestimmung

„ *Most good programmers do programming not because they expect to get paid or get adulation by the public, but because it is fun to program.*

— **Linus Torvalds**

(Finnish American, software engineer and hacker)

Um das allgemeine Verständnis zu gewährleisten müssen noch einige wichtige Begriffe geklärt werden, die die Grundlage für die folgende Untersuchung bilden.

Die Entscheidung, welche der mobilen Variante man für die Umsetzung verwendet ist abhängig von den jeweiligen Eigenschaften. Auf wieviel Speicher darf die App zugreifen, welche Handfunktionen werden benötigt und lässt sich feststellen wann eine Internetverbindung besteht und lassen sich davon abhängig die Daten mit dem Server abgleichen.

2.1 Native Apps

Die Nativen Apps werden speziell für das jeweilige Betriebssystem entwickelt z.B. iOS oder Android. Diese laufen dann auch ausschließlich auf iOS Geräten wie dem iPhone und dem iPad, oder Android Geräten wie dem Samsung Galaxy S4.[@App14]

Dadurch stellt man eine optimale Nutzung der Ressourcen und einheitlich funktionierende Hardwareschnittstellen sicher.[@App14]

2.1.1 Vorteile

- Native Apps nutzen die Leistung des Betriebssystems und des verwendeten Gerätes voll aus, da sie speziell für das Betriebssystem angepasst sind. Dadurch lassen sich sehr gut komplexere und rechenintensivere Apps umsetzen.[@App14]
- Durch die Installation der Apps auf dem Endgerät, können Hardwarefunktionen wie Kamera, Beschleunigungssensor oder **GPS!**¹ benutzt werden. Das ist in der Regel nur nativen Apps vorbehalten.[@App14]
- Daten können auf dem Endgerät in beliebiger Menge gespeichert werden.[@App14]

¹GPS!

- Da Native Apps über einen Appstore vertrieben werden, werden diese öfter gekauft, wenn Sie über gute Bewertungen verfügen.[@App14]
- Die App lässt sich sehr einfach über den Appstore installieren und es wird automatisch ein Icon zum Starten angelegt.[@App14]
- Der Vertriebsaufwand ist sehr gering, da die Appstores verbreitete Bezugsquellen für Native Apps sind. Ist die App erfolgreich, kann man sich in den Top-Listen der App Stores wiederfinden und dadurch sehr hohe Downloadzahlen erreichen.[@App14]

2.1.2 Nachteile

- Ein großer Nachteil ist der erforderliche Entwicklungsaufwand, wenn man die App in allen Appstores anbieten möchte. Dafür muss man die App an die jeweiligen Gegebenheiten des Betriebssystems optimieren.[@App14]
- Es entstehen zusätzliche Kosten um die App für den entsprechenden entwickeln und anbieten zu können.

2.2 Webapps

Die sogenannten Webapps sind im eigentlichen Sinne speziell programmierte **HTML5!**² Websites, die erkennen auf welchem Endgerät sie aufgerufen werden und optimieren den Inhalt entsprechend. Somit kann quasi jedes mobile Endgerät, dass über einen Webbrowser verfügt, die App nutzen.

2.2.1 Vorteile

- Web Apps sind quasi unabhängig vom Betriebssystem und funktionieren auf allen Smartphones. Dadurch erreicht man mehr potentielle Nutzer, bei gleichzeitig geringeren Kosten.[@App14]
- In der Regel kommt man mit der Entwicklung einer Web App günstiger, als mit der Entwicklung einer nativen App für nur ein Betriebssystem.[@App14]
- Durch die Verwendung von HTML5 wird auch die Offline-Speicherung von Daten ermöglicht. Somit kann man auch ohne permanente Internetverbindung die einmal geladene Web App nutzen.[@App14]
- Über Onlinesuchmaschinen wie z.B. Google können Web Apps ohne großen Aufwand gefunden werden und lassen sich auch ohne Installation direkt nutzen. Speichert man diese als Lesezeichen, lässt sie sich genau wie eine Native App vom Startbildschirm aus starten.[@App14]

²HTML5!

- Die Veröffentlichung und Aktualisierung erfolgt in Sekundenschnelle, da sie im Gegensatz zu Native Apps keinen Zulassungsprozess durchlaufen müssen.[@App14]
- Vertriebt man die App selbst, entfällt die Provision von üblicherweise 30% an den Betreiber des App Stores.[@App14]
- Hat man vor die irgendwann die Vorteile einer nativen App zu nutzen und beachtet das bei der Programmierung der Web App, lässt sich diese leicht und kostengünstig in eine Native App umwandeln.[@App14]

2.2.2 Nachteile

- Die meisten Hardwarefunktionen der mobilen Geräte lassen sich gar nicht oder nur mit spezieller Zustimmung des Nutzers verwenden.[@App14]
- Komplexe Berechnungen wie z.B. 3D Darstellungen, Verschlüsselung oder Bildbearbeitungen sind mit einer Web App nicht möglich.[@App14]
- Benötigt die App mehr als 10MB an Datenmaterial auf dem Endgerät ist von einer Entwicklung als reine Web App abzusehen.[@App14]
- Geschäftsmodelle die auf In-App-Käufe oder einen App Store aufbauen, funktionieren zusammen mit der Web App nicht.[@App14]

2.3 Hybridapps

Hybridapps sollen die Vorteile der Web App Entwicklung und der Entwicklung von nativen Apps in sich vereinen. Dabei setzen die Entwickler auf eine große Anzahl von Frameworks. PhoneGap, Corona oder Appcelerator Titanium sind Beispiele dafür, mit deren Hilfe man Web Apps in eine native App umwandeln kann.

Für Einige mag die Entwicklung einer Hybridapp als "Allheilmittel" klingen, jedoch gibt es auch hier Vor- und Nachteile.[@App14]

2.3.1 Vorteile

- Durch die Verwendung einer Hybrid App lässt sich eine Cross Browser Web App erstellen, die in allen modernen Browsern läuft.[@App14]
- Da eine Web App mittels Frameworks für verschiedene Betriebssysteme umgewandelt werden kann, erspart man sich die eigenständige Entwicklung für jedes einzelne Betriebssystem. Es bleiben im schlimmsten Fall nur Betriebssystemspezifische Feinheiten, die noch angepasst werden müssen.[@App14]

- Mit Javascript lassen sich viele Hardwarefunktionen der Endgeräte nutzen, auf die man bei einer Web App nicht zugreifen konnte.[@App14]
- Der Verkauf einer Hybrid App kann wieder über den jeweiligen App Store erfolgen.[@App14]

2.3.2 Nachteile

- Ein großer Nachteil der Hybridapps kann entstehen, wenn man sehr rechenintensive Anwendungen verwendet. Dadurch können Hybrid Apps sehr schnell an das Leistungsmaximum heranreichen und träge reagieren. Das ist sehr stark von dem verwendeten Framework abhängig und ein Nachteil der in Zukunft durch merklich effizienter werdende Frameworks behoben werden kann.[@App14]
- Die Programmierung einer Hybrid App könnte mit zunehmendem Komplexitätsgrad sehr aufwendig werden und eine Umsetzung mittels nativer App empfehlenswerter machen.[@App14]

2.4 Wahl der Appvariante

Die Wahl der Appvariante ist nicht allein abhängig von den Vor- und Nachteilen der Varianten. Hierfür muss man auch die bestehende Website betrachten und die Umsetzungsaufwände gegeneinander abwägen.

Ein nicht zu unterschätzender Vorteil der Website ist die bereits responsive Umsetzung. (Abb.2.1-3.2) Dadurch passt sich der dargestellte Inhalt an das Endgerät an, wodurch optimale Darstellung auch auf mobilen Geräten gewährleistet wird. Es können somit bereits neue Daten mobil erfasst werden.

Thüringen
erleben.de

Einrichtungen

Administration

Einrichtungen suchen ...

hantschack@webit.de

Erfolgreich angemeldet

Einrichtungen

← zurück

1

2

weiter →

Bezeichnung	Buchungscode	Adresse	Formulare	Letzte Änderung
Hotel "Zur Erholung"	EIC46009	Brochthäuserstr. 65 37115 Duderstadt	7	vor 4 Tagen
Ferienhaus Altes Schlächthaus		Schmiedefelder Straße 6 98711 Vesser	15	vor 8 Tagen
co.med - Das Gesundheitshotel ****	SLF31539	Rainweg 68 b 07318 Saalfeld	33	vor 8 Tagen
City-Apartments Mühlhausen	MHL45666	Untermarkt 12 99974 Mühlhausen / Th	15	vor 8 Tagen
Hotel Thüringen Suhl	CCS62010	Platz der Deutschen Einheit 2 98527 Suhl	27	vor 10 Tagen
Hotel an der Therme - Haus 3	WMR47264	Rudolf-Gröschner-Str. 11 99518 Bad Sulza	26	vor 17 Tagen
Hotel Goldener Hirsch	CCS62005	An der Hasel 91-93 98527 Suhl-Neundorf	12	vor 22 Tagen
Ringberg Hotel	CCS62007	Ringberg 10 98527 Suhl	74	vor 23 Tagen
egapark			0	vor 23 Tagen
Bio - Seehotel Zeulenroda	GRZ33026	Rudolf-Gröschner-Str. 11 99518 Bad Sulza	31	vor 25 Tagen

← zurück

1

2

weiter →

+

Neue Einrichtung anlegen

Abb. 2.1: Barrierefreiheitsdatenbank

Thüringen
erleben.de

Einrichtungen

Administration

Einrichtungen suchen ...

hantschack@webit.de

Neue Einrichtung anlegen

Bezeichnung

Bitte eintragen

TomasID

Bitte eintragen, falls vorhanden

OpenTextID

Bitte eintragen, falls vorhanden

Buchungscode

Bitte eintragen, falls vorhanden

Beschreibung

Tragen Sie hier auf Wunsch eine Beschreibung ein

Art der Einrichtung

☐ Beherbergungsbetrieb
☐ Gastronomie
☐ Bahnhof
☐ Shopping
☐ Freizeiteinrichtung

Zertifikate

Bitte wählen Sie die passenden Zertifikate

Speichern

Abbrechen

Abb. 2.2: Datenerfassung

Einrichtungen

← zurück 1 2 weiter →

Bezeichnung	Buchung
Hotel "Zur Erholung"	EIC4600
Ferienhaus Altes Schlachthaus	
co.med - Das Gesundheitshotel ****	SLF315
City-Apartments Mühlhausen	MHL456
Hotel Thüringen Suhl	CCS620
Hotel an der Therme - Haus 3	WMR470
Hotel Goldener Hirsch	CCS620
Ringberg Hotel	CCS620
egapark	
Bio - Seehotel Zeulenroda	GRZ330

← zurück 1 2 weiter →

Neue Einrichtung anlegen

Neue Einrichtung anlegen

Bezeichnung

Bitte eintragen

TomasID

Bitte eintragen, falls vorhanden

OpenTextID

Bitte eintragen, falls vorhanden

Buchungscode

Bitte eintragen, falls vorhanden

Beschreibung

Tragen Sie hier auf Wunsch eine Beschreibung ein

Art der Einrichtung

- ☐ Beherbergungsbetrieb
- ☐ Gastronomie
- ☐ Bahnhof
- ☐ Shopping
- ☐ Freizeiteinrichtung

Zertifikate

Bitte wählen Sie die passenden Zertifikate

Speichern [Abbrechen](#)

Abb. 2.3: Barrierefreiheitsdatenbank mobil

Nach Betrachtung der Vor- und Nachteile der App Varianten und der bestehenden Website, schließe ich die Umsetzung einer reinen nativen App aus. Der Umsetzungsaufwand und die entstehenden Kosten wären unangemessen hoch im Vergleich zu den daraus entstehenden Vorteilen.

Da eine Web App auf **HTML5!** aufbaut und die Website sich bereits teilweise an das Endgerät anpasst, baut die Untersuchung im auf eine Umsetzung als Web App auf. Daher wird bei der Synchronisation auf **HTML5!** und Javascript gesetzt.

Synchronisation

“ *The most important property of a program is whether it accomplishes the intention of its user.*

— C.A.R. Hoare

(British computer scientist, winner of the 1980 Turing Award)

Synchronisation beschreibt den Datenaustausch zwischen einem Sender und einem Empfänger. Dabei werden die Daten in Blöcke aufgeteilt und in einen Übertragungsrahmen eingepasst. Um die Daten aneinander anzugleichen muss dabei festgestellt werden, welches Endgerät welche Daten besitzt und kontrolliert ob das andere Gerät diese Daten zu seinen besitzen will.

Besitzen beide Endgeräte dieselben Daten, z.B. in unterschiedlichen Versionen, kann definiert werden, wie mit den Änderungen umgegangen wird.[@Ope13]

Durch die Begrenzung des lokalen Gerätespeichers würde ich die Synchronisation auf 2 separaten Wegen durchführen. Im ersten Teil die Datenbank und als Zweites alle Anhänge, wie z.B. Bilder oder Dokumente. So kann das Problem der Speicherbegrenzung bei Webapps umgangen werden.

3.1 Datenbanksynchronisation am Bsp. WebSqlSync

WebSqlSync ist eine Javascript Bibliothek zur automatischen Synchronisation einer lokalen WebSql Datenbank mit dem Server. Die Synchronisation kann dabei in beide Richtungen erfolgen und arbeitet auf dem Prinzip der inkrementellen Synchronisation, was bedeutet dass nur erforderliche Daten übertragen werden.

WebSqlSync funktioniert auch ohne Internetverbindung. Alle Änderungen der Daten werden dabei verfolgt und mit dem Server abgeglichen, sobald wieder eine Internetverbindung besteht. Es wird auch die Änderung auf mehreren Geräten unterstützt.

Die Unterstützung von webapp und der phonegap app für mobile Betriebssysteme wie z.B. iOS und Android ermöglicht eine einfache Integration ohne den Programmcode anpassen zu müssen.[@orb13]

Installation und Initialisierung

Um WebSqlSync nutzen zu können, muss nur die Datei webSqlSync.js im **HTML!**¹ des Projekts hinzugefügt werden.

```
<script src="lib/webSqlSync.js" type="application/x-javascript" charset="utf-8"></script>
```

Wird die Bibliothek aufgerufen, werden automatisch 2 Datenbanktabellen erstellt, falls diese nicht bereits von einem vorherigen Aufruf existieren. Die erste Tabelle **new_elem** speichert alle neuen bzw. geänderten Elemente und die zweite Tabelle **sync_info** das Datum der letzten Synchronisation.

Zusätzlich werden sogenannte SQLite Auslöser erstellt, die überwachen ob Änderungen per **INSERT** oder **UPDATE** an den Tabellen vorgenommen wird. SQLite ist eine einfache Datenbankbibliothek die Befehle der Sprache **SQL!**² verwendet.

Geänderte Elemente werden somit automatisch in der Tabelle **new_elem** eingefügt.

```
DBSYNC.initSync(  
    TABLES_TO_SYNC, webSqlDb, sync_info ,  
    'http://www.myserver.com', callBackEndInit  
);
```

Die Tabellen die man mit dem Server synchronisieren möchte, werden in der Funktion **TABLES_TO_SYNC** angegeben.

```
TABLES_TO_SYNC = [  
    {tableName : 'table1', idName : 'the_id'},  
    {tableName : 'table2'}  
    //if idName not specified, it will assume that it's "id"  
];
```

In der Tabelle **sync_info** können alle Informationen gespeichert werden, die der Entwickler als nützlich empfindet. Die Identifikation des Clients wäre eine wichtige Eigenschaft, da Sie mit an den Server gesendet wird. Dafür kann jegliche Information genutzt werden, wie z.B. die Emailadresse, ein Login oder auch eine entsprechende **ID!**³ des genutzten mobilen Endgeräts.

¹HTML!

²SQL!

³ID!

Aufruf

Um die Synchronisation zu starten ruft man die Funktion **syncNow** auf. Die Synchronisation kann dabei nach einer freiwählbaren Zeitspanne, oder aber nach einer festgelegten Anzahl von Datenänderungen erfolgen.

```
DBSYNC.syncNow(callBackSyncProgress, function(result) {
    if (result.syncOK === true) {
        //Synchronized successfully
    }
});
```

Bei größeren Datenmengen ist es für den Nutzer hilfreich, wenn man eine Fortschrittsanzeige bekommt. Während der Synchronisation wird dafür bei jedem Einzelschritt die Funktion **callBackSyncProgress** aufgerufen.

```
callBackSyncProgress: function(message, percent, msgKey) {
    $('#uiProgress').html(message + ' (' + percent + '%)');
},
```

Einschränkungen

Die Bibliothek WebSqlSync hat auch ein paar wenige Einschränkungen. Z.B. wird der **SQL!**-Befehl **DELETE** nicht unterstützt. Stattdessen sollte das mit einem update an der entsprechenden Stelle umgangen werden.

3.2 Dateisynchronisation am Bsp. ownCloud

ownCloud ist eine Open Source Software für die Einrichtung einer unabhängigen Serverseitigen Datenspeicherlösung. Sogenannte Cloudspeicher gibt es mittlerweile sehr viele. Die wohl bekanntesten sind Dropbox, Google Drive und OneDrive. Sie ermöglichen einen einfachen Datenzugriff von überall auf der Welt. Dafür lädt man seine Dateien über eine Internetverbindung auf einen speziellen Datenserver. Leider hat das den großen Nachteil, dass man keine 100 prozentige Sicherheit hat, was mit den eigenen Daten passiert.

Um das zu sicherzustellen bietet ownCloud eine Möglichkeit einen eigenen Cloudserver zu erstellen, auf dem man alles selbst konfigurieren kann. Per Nutzer- und Rechteverwaltung lässt sich wie auf einem normalen Server festlegen, wer was machen kann.

Dazu werden Backup Lösungen angeboten, so dass jederzeit Sicherungen der Dateien angelegt werden können. Der wohl wichtigste Punkt ist aber die Verschlüsselung der gespeicherten Daten mittels **SSL!**⁴, somit können auch unternehmensrelevante Daten gespeichert werden.

⁴SSL!

Durch die Open Source Lösung, lässt sich zu den bereits vorhandenen Plugins, neue Software entwickeln und den eigenen Bedürfnissen anpassen. Da bereits Apps für die mobilen Betriebssystem zu Verfügung stehen, lassen sich relativ einfach Daten zwischen dem Server und dem Client synchronisieren.

Neben der frei verfügbaren “Community Edition” stehen für die volle Unterstützung der ownCloud Entwickler auch eine “Business” und eine “Enterprise Version” zur Verfügung, welche gegen eine jährliche Gebühr angeboten wird.[@Fra14]



Abb. 3.1: Architektur ownCloud

Vorteile Android/iOS App

- SSL- und HTTP-Verbindungen werden automatisch erkannt, so dass eine einfache und gesicherte Verbindung zu dem Server möglich wird.[@Fra14]
- Dateien und Ordner auf dem Server können neu angelegt, durchsucht, umbenannt und gelöscht werden, je nachdem wie die Rechte an den Nutzer vergeben sind.[@Fra14]
- Durch das Anlegen von Favoriten lassen sich Daten wie Dokumente, Bilder und Videos automatisch mit dem Server synchronisieren.[@Fra14]

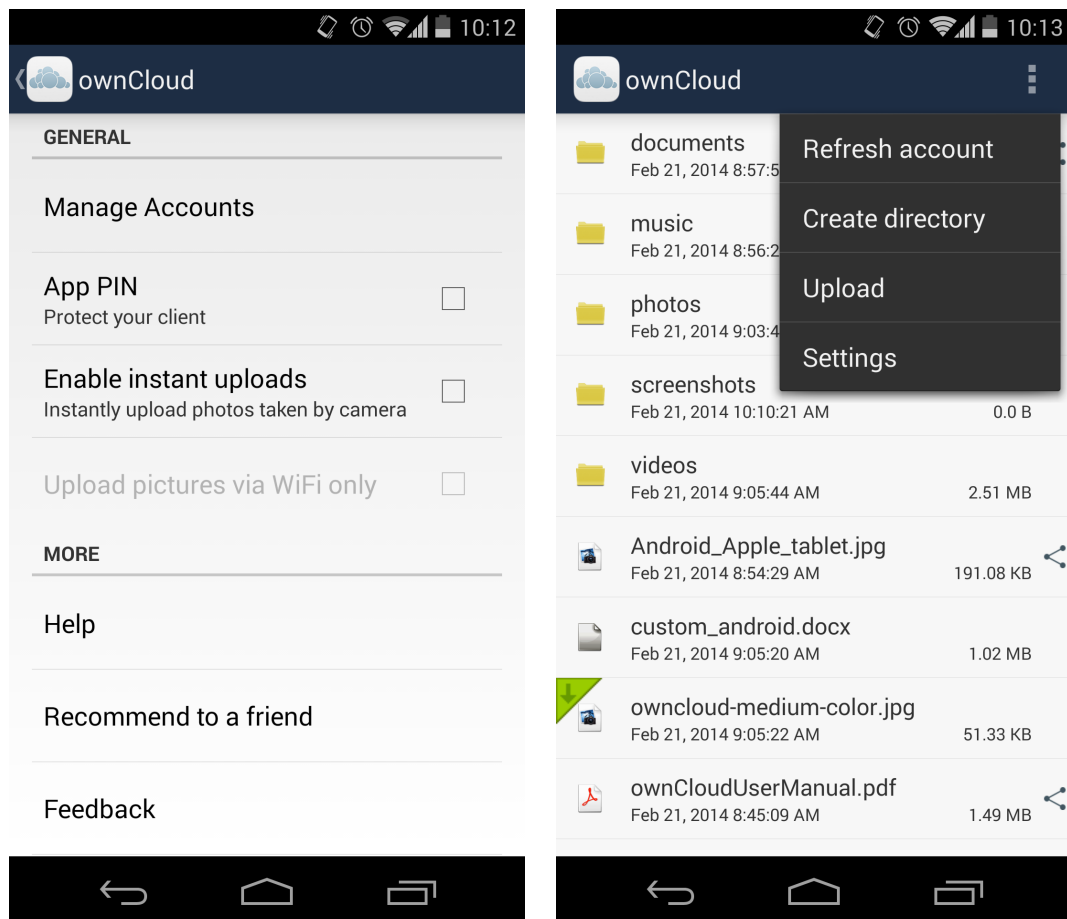


Abb. 3.2: ownCloud App

Fazit

4.1 System Section 1

4.2 System Section 2

4.3 Ausblick

Webseiten

- [@App14] App Entwickler Verzeichnis. *7,3 Millionen schwerbehinderte Menschen*. 2014. URL: <http://www.app-entwickler-verzeichnis.de/faq-app-entwicklung/11-definitionen/107-unterschiede-und-vergleich-native-apps-vs-web-apps> (besucht am 2. Mai 2014) (zitiert auf den Seiten 4–7).
- [@Fra14] Frank Karlitschek. *ownCloud*. 2014. URL: <http://owncloud.org/> (besucht am 5. Mai 2014) (zitiert auf Seite 14).
- [@Hyp10] HyperJoint GmbH. *DIN-18040*. 2010. URL: <http://nullbarriere.de/din18040-1.htm> (zitiert auf Seite 1).
- [@Ope13] Open Mobile Alliance. *SyncML Spezifikationen*. 2013. URL: http://technical.openmobilealliance.org/Technical/release_program/SyncML_v1_2_2.aspx (besucht am 3. Mai 2014) (zitiert auf Seite 11).
- [@Sta11] Statistisches Bundesamt Wiesbaden. *7,3 Millionen schwerbehinderte Menschen*. 2011. URL: <https://www.destatis.de/DE/ZahlenFakten/GesellschaftStaat/Gesundheit/Behinderte/Aktuell.html> (besucht am 27. Mai 2011) (zitiert auf Seite 1).
- [@orb13] orbitaloop. *WebSqlSync*. 2013. URL: <http://www.verious.com/code/orbitaloop/WebSqlSync/> (besucht am 3. Mai 2014) (zitiert auf Seite 11).
- [@zef14] zefhemel. *persistence.js*. 2014. URL: <https://github.com/zefhemel/persistencejs> (besucht am 5. Mai 2014).

Abbildungsverzeichnis

2.1	Barrierefreiheitsdatenbank	8
2.2	Datenerfassung	8
2.3	Barrierefreiheitsdatenbank mobil	9
3.1	Architektur ownCloud	14
3.2	ownCloud App	15

Selbstständigkeitserklärung

Ich, Christian Schramm, Matrikel-Nr. s3001102, versicher hiermit, dass ich meinen Praxistransferbeleg mit dem Thema

Barrierefreiheitsdatenbank - Untersuchung ob es eine Möglichkeit gibt Daten und Bilder auf mobilen Endgeräten offline zu Erfassen und bei bestehender Internetverbindung mit dem Server abzugleichen

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Dresden, April 8, 2014

Christian Schramm

