Überleben im Funkloch

Offlinefähige Apps für Android entwickeln

Christian Janz (@c_janz)

christian.janz@bridging-it.de

Christian Janz

Consultant im Bereich Softwareentwicklung Java/JEE bei
Bridging IT in Mannheim

Interesse: Architektur und Entwicklung von
Geschäftsanwendungen mit Hilfe moderner Java Frameworks

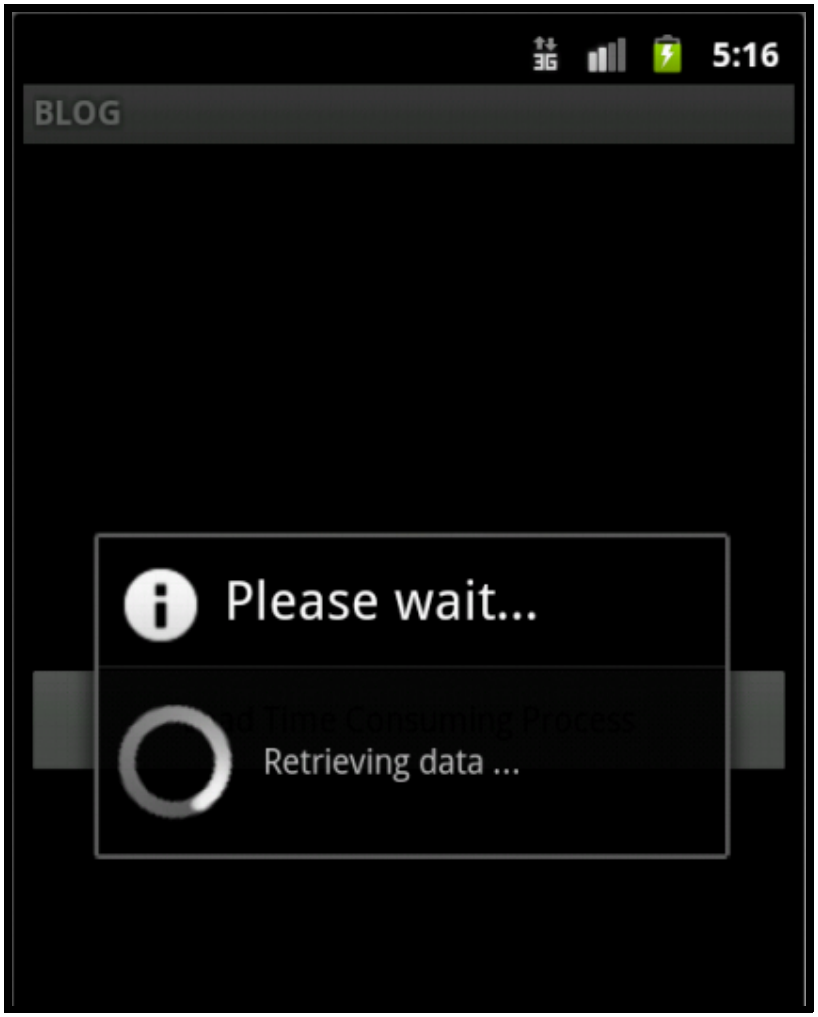
Twitter: **@c_janz** | E-Mail: **christian.janz@bridging-it.de**
Slides: **http://de.slideshare.net/cjanz**

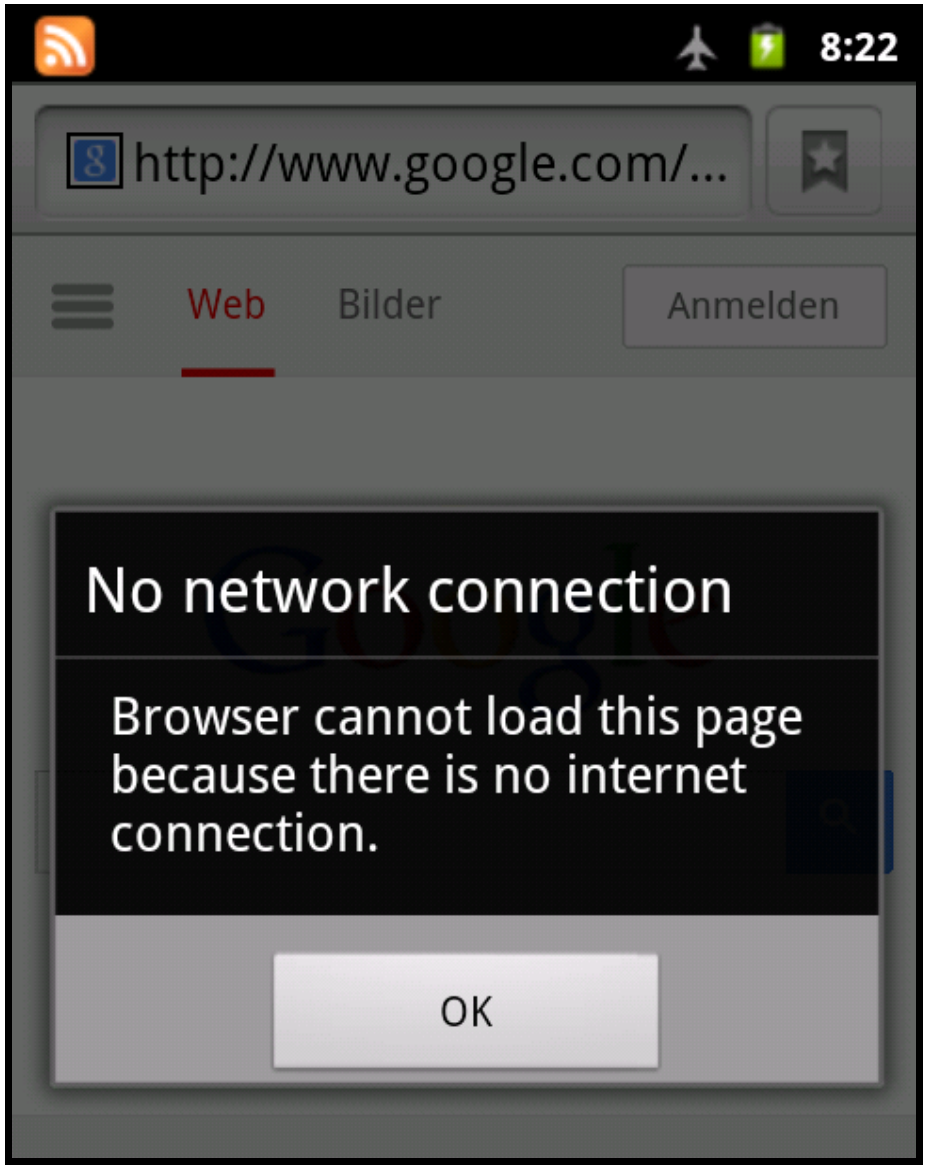
Agenda

- Motivation
- Lösungsidee
- Architekturansatz
- APIs im Einsatz: Sync Sample
- Fazit

Motivation


DB
ICE


5:16
BLOG
Please wait...
Retrieving data ...


8:22
http://www.google.com/...
Web
Bilder
Anmelden
No network connection
Browser cannot load this page because there is no internet connection.
OK


10:50
All Items
NEW ITEMS
Project Euler 140: Modified Fibonacci golden nuggets - mathblog.dk
Theoretische Informatik - Wikipedia - Beobachtungsliste [de]
Hesse-Matrix - Wikipedia - Beobachtungsliste [de]
Hesse-Matrix - Wikipedia - Beobachtungsliste [de]
Help Desk Hangouts
and Page Update
Shadow DOM 201
CSS and Styling -
HTML5Rocks - Posts & Tutorials
Stable Channel Update for Chrome OS -
Google Chrome Releases
Prisma Media Saves 30 percent in 45 Days with Google Apps - The Google Apps Blog
Color-Seeing Machine Sorts Your Sweet
STOP!

1. Juli 2013

Lösungsidee

Synchronisiere Daten und speichere sie lokal auf dem Gerät

Vorteile

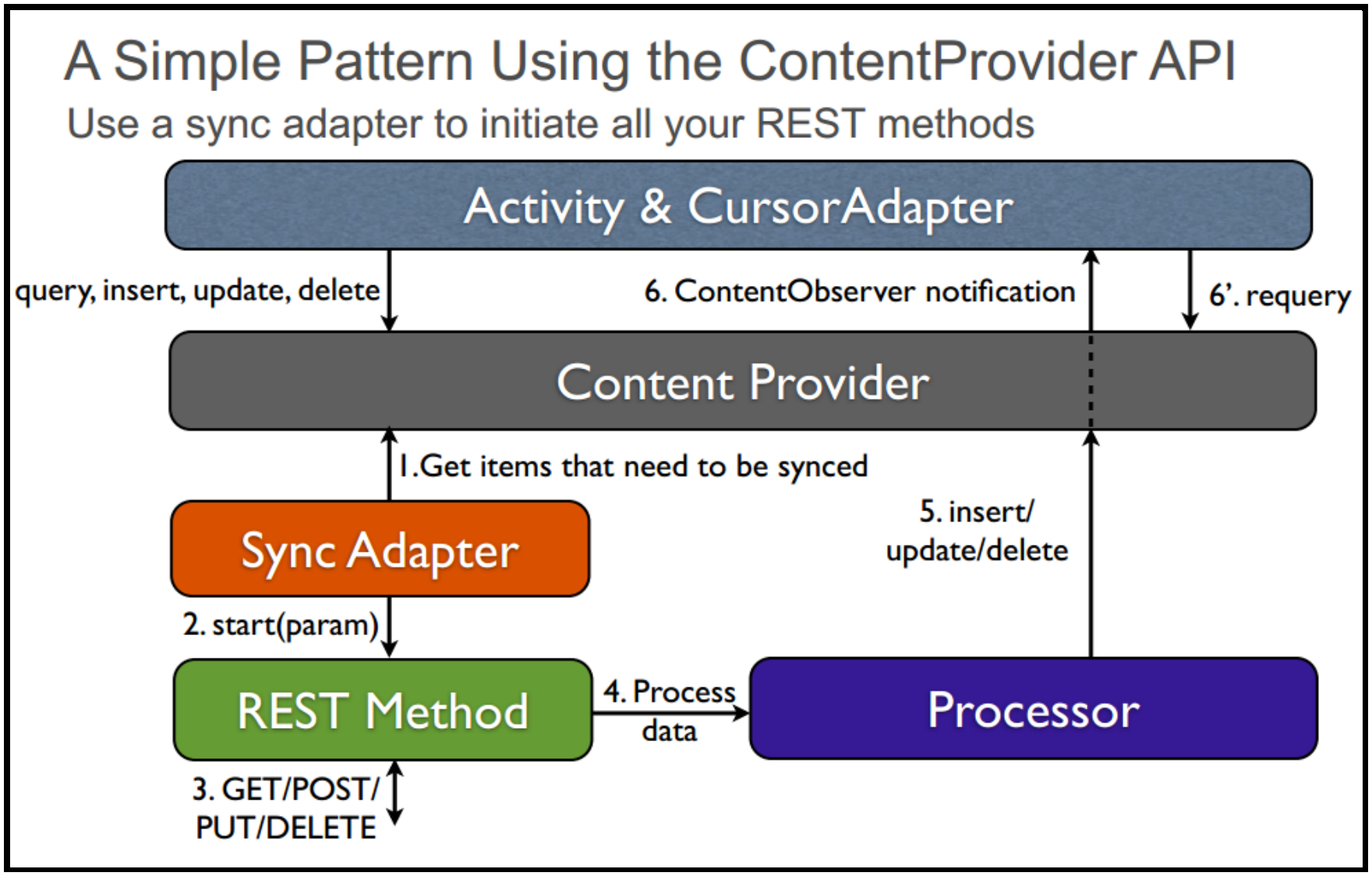
- App kann auch ohne aktive Datenverbindung genutzt werden
- Datenvolumen wird reduziert
- Akkulaufzeit wird erhöht
- Daten werden automatisch aktualisiert, wenn eine Datenverbindung besteht
- Retry bei abgebrochener Verbindung
- "Nebenbei": Verbesserte Architektur

Architektur

Architekturansatz für Android

Vortrag von Virgil Dobjanschi auf der Google IO 2010

"Developing Android REST Client Applications"



Quelle: Developing Android REST Client Applications

SyncAdapter: Features

- Beachten von Netzwerkverfügbarkeit
- Synchronisation auch wenn App nicht läuft
- Periodische Synchronisation
- Warteschlage
- Bündeln von Syncs: Gut für Akkulaufzeit
- Integration in globale Sync-Einstellungen
- Integration in Account-Verwaltung

APIs im Einsatz: Sync Sample

Vorgehen

- Authenticator & Account
- ContentProvider
- REST client
- SyncAdapter
- UI

https://github.com/cjanz/android-sync-sample

Authenticator & Account

- AuthenticatorService
- Authenticator
- LoginActivity
- authenticator.xml
- AndroidManifest.xml

authenticator.xml

```
<account-authenticator
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:accountType="de.bit.android.sample.account"
        android:icon="@drawable/ic_launcher"
        android:smallIcon="@drawable/ic_launcher"
        android:label="@string/app_name"
/>
```

AndroidManifest.xml

```
<service
    android:name="de.bit.android.syncsample.authenticator.AuthenticatorService" >
    <intent-filter>
        <action android:name="android.accounts.AccountAuthenticator" />
    </intent-filter>

    <meta-data
            android:name="android.accounts.AccountAuthenticator"
            android:resource="@xml/authenticator" />
</service>

<activity
        android:name="de.bit.android.syncsample.authenticator.LoginActivity"
        android:excludeFromRecents="true"
        android:exported="true"
        android:theme="@android:style/Theme.Holo" >
</activity>
```

Demo: Authenticator & Account

ContentProvider

- TodoContentProvider
- DatabaseHelper
- TodoEntity
- AndroidManifest.xml

TodoEntity

```
public class TodoEntity {
    private Long id;
    private Long serverId;
    private Long serverVersion;
    private Long conflictedServerVersion;
    private SyncState syncState = SyncState.NOOP;

    private String title;
    private String text;

    ... (getters and setters)
}
```

REST client

```
public class TodoRestClient {

    public static List<TodoEntity> loadAllTodos() throws IOException, JSONException;

    public static TodoEntity saveTodo(TodoEntity todoEntity) throws IOException, JSONE

    public static void deleteTodo(TodoEntity todoEntity) throws IOException;
}
```

SyncAdapter

- SyncService
- SyncAdapter
- syncadapter.xml
- AndroidManifest.xml

syncadapter.xml

```
<?xml version="1.0" encoding="utf-8"?>
<sync-adapter
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:accountType="de.bit.android.sample.account"
        android:contentAuthority="de.bit.android.syncsample.content"
        android:supportsUploading="true" />
```

AndroidManifest.xml

```
<service android:name="de.bit.android.syncsample.sync.SyncService" >
    <intent-filter>
        <action android:name="android.content.SyncAdapter" />
    </intent-filter>

    <meta-data
            android:name="android.content.SyncAdapter"
            android:resource="@xml/syncadapter" />
</service>
```

Demo: Speichern der Daten vom Backend

```
@Override
public void onPerformSync(Account account, Bundle extras, String authority,
                          ContentProviderClient provider, SyncResult syncResult) {
                              ...
}
```

Periodische Synchronisation

```
private void configureSync(Account account) {
    ContentResolver.setIsSyncable(account, CONTENT_AUTHORITY, 1);
    ContentResolver.setSyncAutomatically(account, CONTENT_AUTHORITY, true);

    Bundle params = new Bundle();
    params.putBoolean(ContentResolver.SYNC_EXTRAS_EXPEDITED, false);
    params.putBoolean(ContentResolver.SYNC_EXTRAS_DO_NOT_RETRY, false);
    params.putBoolean(ContentResolver.SYNC_EXTRAS_MANUAL, false);
    ContentResolver.addPeriodicSync(account, CONTENT_AUTHORITY, params, 60);

    ContentResolver.requestSync(account, CONTENT_AUTHORITY, params);
}
```

UI

- MainActivity
- activity_main.xml
- todo_row.xml
- EditTodoActivity
- activity_edit_todo.xml
- AndroidManifest.xml

Demo: Sync-Status anzeigen

```
@Override
protected void onResume() {
    super.onResume();

    syncObserverHandle = ContentResolver.addStatusChangeListener(
            SYNC_OBSERVER_TYPE_ACTIVE | SYNC_OBSERVER_TYPE_PENDING, this);
}

@Override
public void onStatusChanged(int which) {
    runOnUiThread(new Runnable() {

        @Override
        public void run() {
            boolean isSyncActive = ContentResolver.isSyncActive(account,
            TodoContentProvider.AUTHORITY);
            setProgressBarIndeterminateVisibility(isSyncActive);
        }
    });
}
```

Demo: Synchronisation anstoßen

```
Bundle params = new Bundle();
params.putBoolean(ContentResolver.SYNC_EXTRAS_EXPEDITED, true);
params.putBoolean(ContentResolver.SYNC_EXTRAS_MANUAL, true);

ContentResolver.requestSync(account, TodoContentProvider.AUTHORITY,
params);
```

Demo: Automatisches Update der UI

CursorLoader und ContentResolver.notifyChange(...)

Demo: Create, Update, Delete

Fazit

- Offlinefähige Apps haben Vorteile
- Offlinefähigkeit muss nicht aufwändig sein
- Android bietet gute Unterstützung dafür
- Dokumentation nicht optimal
- Fehlerhandling muss beachtet werden

Links

- **Transferring Data Using Sync Adapters | Android Developers**
- **Slides: Android SyncAdapter | Alex Tumanoff**
- **Tutorial: Write your own Android SyncAdapter | Udinic**
- **SyncAdapter Sample App | Christian Janz**

Fragen?

ENTWICKLERTAG

meet the SPEAKER @speakerlounge

1. OG DIREKT ÜBER DEM EMPFANG