Surviving a Successful App Launch

Siegfried Goeschl, Mathaeus Zingerle
• Leading Austrian classified ads provider
• 2,2 million adverts online
• 3,8 million unique clients/month
• 17,2 million visits/month
• 583,5 million page impressions/month

162.800 downloads in the first two weeks
612,660 downloads in 5 months

iOS & Android 01.07. - 30.11.2013
Launching July 2013

- Why not July 2008?!
- Using Java/JSP-centric platform
- Complex code base after 10 years
- Lots of business logic in JSPs & taglibs
- No “out-of-the-box” REST API
No REST API
No Mobile Apps
The REST API

- Sits below UI module dependency-wise
- Exposes simplified business logic
  - Mixture of refactoring and rewriting
- Using Jersey REST Service stack
- Three months development time for v1.0
REST API Guidelines

- Think of human-browsable REST API
- Smart API and stupid apps
- REST API bootstrapping
- REST API versioning
Human-Readable API

```
<selfLink>
  http://api.willhaben.at/restapi/v2/search/ads/5/301?sort=1&…
</selfLink>
```

versus

```
<atom:link
  href="http://api.willhaben.at/restapi/v2/search/ads/5/301?
  sort=1&…"
  rel="self"
  type="text/xml"
  title="Current search"
/>
```
Smart API & Stupid Apps

- Apps shall be using and not building URLs
  - Building URLs requires knowledge
- If in doubt implement logic in the API
  - You can’t update mobile apps easily
  - And much easier to test
REST API Bootstrapping

```xml
<?xml version="1.0" encoding="UTF-8"?>
<vendors>
  <vendor name="willhaben">
    <services>
      <service name="remoteapi">
        <environments>
          <environment name="dev">
            <versions>
              <version major="2" minor="0">
                <rootUri>http://api-dev.willhaben.at/restapi/v2</rootUri>
              </version>
            </versions>
          </environment>
          <environment name="prod">
            <versions>
              <version major="1" minor="0">
                <rootUri>http://api.willhaben.at/restapi/v1</rootUri>
              </version>
              <version major="2" minor="0">
                <rootUri>http://api.willhaben.at/restapi/v2</rootUri>
              </version>
            </versions>
          </environment>
        </environments>
      </service>
    </services>
  </vendor>
</vendors>
```
REST API Versioning

• You designed and implemented the latest & greatest API V1.0
  ▶ It is extendable
  ▶ It is scalable
  ▶ It is [you name it]
• You will write a V2.0
REST API Versioning

- Need to support V1.x and V2.x in parallel
- Used backward compatibility layer before
  - Complicated and fragile
- Using a different approach for now
  - Copy & Pasta API versioning
Copy & Pasta Versioning

- Version identifier in the package name
- Copied *.v1.* to *.v2.* packages
  - Now you have different implementations which makes Spring & Jersey happy
  - Allows to use “v2” URLs and XML namespaces
  - Only critical bugfixes are done in V1.x
Now we have an API but still no apps?!
Android & iOS Apps

- No internal know-how
- Building internal know-how takes time
- Teaming up with Tailored Apps
One Day Workshop

- Working together in one room
- Writing Android & iOS prototype
- Using the shiny new REST API v1.0
- Prototype must be running on a real phone at the end of the day
Android Prototype
BMW 3er-Reihe 330d xD... BILDER

EXTERIOR_COLOUR_MAIN: Grau

DESCRIPTION: Garantie: Garantie-/Gewährleistungszeit: 12 Monat(e)
KFZ Typ: 330d xDrive Touring
Österreich-Paket Aut.
Anzahl Sitze: 5
Genaue Farbezeichnung: spacegrau metallic
Innenfarbe: Stoff vertex anthrazit
Hubraum: 2993 ccm

Ausstattung Detail:
Ablagenpaket Airbag für Fahrer und Beifahrer Allradsystem xDrive, mit variabler Drehmomentverteilung Armauflage vorne und hinten, in Türverkleidung integriert Armauflage vorne, verschiebbar
Mathäus is speaking
- Product Management and UX Designer -> Sketches, Mockups, Screens
- Product Development got order to create API to offer features of Willhaben website.
- App Developer between PM and PD working with Mockups/Screens and API.
- Weekly iterations to give and get feedback on UI and API.
Test Phase iOS

- Beta builds made available via iOS InHouse Distribution
- Used TestFlight later on
  - User registration required
  - Device registration required
  - Installation of profiles
Test Phase Android

• Using Alpha channel of the Google Play Store to distribute alpha versions internally.

• Friendly users got a link to a APK for beta testing. Easier than Beta channel.

• Similar barriers as with iOS
iOS Components

- Cocoapods
  - AFNetworking
  - Mantle for marshalling/unmarshalling
  - BlocksKit for asynchronous operations
  - Kiwi for TDD
Android Components

- ActionbarSherlock (who doesn't use it?)
- Android Annotations
- Robospice (SpringAndroid/GSON)
- Universal Image Loader
- EndlessAdapter
- GooglePlayServices for location
- Ant, later Gradle
The Power of Open Source

- Memory Leak found before final release
- Eclipse Memory Analyzer points to RoboSpice listener
- No solution on the mailing list
- Forked Robospice and removed listener
- Tests showed the memory leak was gone
- Ship it
Sony Xperia 2.x

Which button to press?!
127 Anzeigen in Wien, 01. Bezirk, Innere Stadt

Schmökern

Liste

- **Luxus Penthouse**
  - 1010 | 192 m²
  - 14.10.13 | €3,500,000

- **146/2690 Fantastische**
  - 1010 | 240 m²
  - 13.10.13 | €2,980,000

- **Bestgelegene Eigentumswohnung,**
Client Time Zones

- Japanese calendar stating the year of the reign of the current emperor - currently Heisei 25 in the year of the snake
- Buddhistic calendar stating 2257
- Usual suspects such as 1970-01-01
Simple Date Format?!

- Using "yyyy-MM-dd'T'HH:mm:ssZZ"

- 2013-07-01T00:18:24+0300 (GT-N7100 Build/JZO54K)

- 2013-08-08T09:00:47:33+0200 (ZTE BLADE III Build/IMM76D)

- 2013-08-29T08:25:00:26+0200 (LT30p Build/9.1.A. 1.141)

- ?????-??-??T???:??+???? (RC0709B Build/IMM76D) Q-Pad
Vendor Distribution

Metric: Visits

- Apple (41.2%)
- Samsung (40.3%)
- Sony (3.4%)
- HTC (3.4%)
- LG (3.3%)
- Sony Ericsson (3.2%)
- Google (1.6%)
- Unspecified (1.2%)
- ZTE (0.4%)
- Asus (0.3%)
- Huawei (0.3%)
- Sony Ericsson (0.2%)
- Others (1.2%)
## Android Versions

<table>
<thead>
<tr>
<th>Version</th>
<th>Count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android 4.1</td>
<td>96,174</td>
<td>47.41%</td>
</tr>
<tr>
<td>Android 2.3.3 - 2.3.7</td>
<td>33,647</td>
<td>16.59%</td>
</tr>
<tr>
<td>Android 4.2</td>
<td>32,611</td>
<td>16.08%</td>
</tr>
<tr>
<td>Android 4.0.3 - 4.0.4</td>
<td>30,200</td>
<td>14.89%</td>
</tr>
<tr>
<td>Android 4.3</td>
<td>6,683</td>
<td>3.29%</td>
</tr>
<tr>
<td>Android 2.2</td>
<td>2,497</td>
<td>1.23%</td>
</tr>
<tr>
<td>Android 3.2</td>
<td>658</td>
<td>0.32%</td>
</tr>
<tr>
<td>Android 3.1</td>
<td>319</td>
<td>0.16%</td>
</tr>
<tr>
<td>Android 2.3</td>
<td>33</td>
<td>0.02%</td>
</tr>
<tr>
<td>Weitere</td>
<td>14</td>
<td>0.01%</td>
</tr>
</tbody>
</table>
How (Not) To Soft Launch

• Development suggests Soft Launch
  ▸ Enough time to fix critical problems
• Product Management opts for Hard Launch
  ▸ Becoming a Top App is mandatory
• Product Management won :-)

How (Not) To Soft Launch

- Allocated additional time for DEV & OPS
- 12-hour performance test based on JMeter with 120 JSON requests/sec
  - Re-tested system failure and recovery scenarios
  - Reviewed system monitoring
  - Updated deployment procedures
We have thousands of testers but only three of them write bug reports

The Problem with Hard Launch
Hard Launch - First Week

• Hundreds of exceptions in the server log file each and every day
  ▶ Tons of users will find any dormant bug
  ▶ First aid patches to work around bugs in the mobile apps

• Releasing twice a day

• No server crashes - no downtime
Now - 5 Months Later
Lessons Learned

• Reading AppStore & PlayStore comments
  ‣ Users are quick to point out problems
  ‣ Feedback for new features

• Monitoring the server logs
  ‣ New app version can cause errors

• REST API 1.x compatibility across multiple app versions is tricky
Lessons Learned

• Use multiple levels of server monitoring
  ▶ Better act than react

• Our favourite tools
  ▶ Java Melody - J2EE monitoring for general server health
  ▶ Java Simon with Spring AOP to monitor the processing on method level
Überwachungsstatistiken von JavaMelody aufgenommen vom 02.12.13 16:02 bis 01.10.12 15:13 auf _web5.willhaben.apa.net

Aktualisieren  Online-Hilfe  Auswahl des Zeitraums: Tag  Woche  Monat  Jahr  Alle  Benutzerdefiniert

Speichernutzung - 1 Tag

- % CPU - 1 Tag

- Http-Sitzungen - 1 Tag

Aktive Threads - 1 Tag

- Aktive JDBC-Verbindungen - 1 Tag

- Verwendete JDBC-Verbindungen - 1 Tag

Http Treffer pro Minute - 1 Tag

- Http mittlere Zeiten (ms) - 1 Tag

- % Http-Fehler - 1 Tag

SQL Treffer pro Minute - 1 Tag

- SQL mittlere Zeiten (ms) - 1 Tag

- % der SQL-Fehler - 1 Tag

- 1,0
- 745
- 1873
- 1,0
- 112
- 1260
- 1,0
- NaN
- NaN
- 1,0
- NaN
- NaN
- 1,0
- NaN
- NaN

Mittel  Maximum  Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum
Mittel:  Maximum

The Future

- Mobile web site based on REST API
- More customer & mobile apps coming
- Using Calabash for E2E testing
- Collecting ideas for REST API V3
Still Awake?!
Questions & Answers
Additional Resources