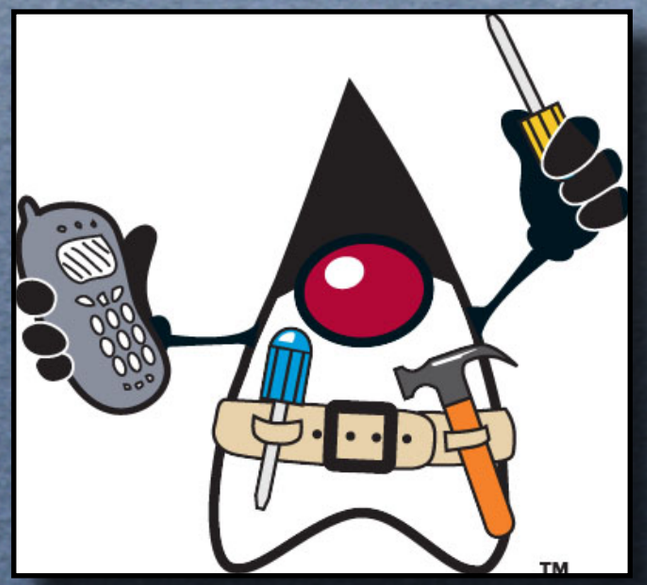
J2ME Software Entwicklung und Fehlersuche

Ein Praxisbericht



Dipl.-Inf.(FH) M.Sc. Michael Kroll

IT-Berater - Michael Kroll Consulting & Solutions

31.05.2007

Was erwartet sie ?

- Überblick über die API Entwicklung
- Praxisbeispiele
  - Anforderungen, Hürden, Lösungen
    - Externe Hilfsmittel
  - Externe APIs
- Keine Debugging Sessions (IDE/Debugger)!

Motivation

- Mobiltelefon mittlerweile so wichtig wie der Haustürschlüssel und die Brieftasche
- Fast jedes neue Mobiltelefon ist Java fähig (*)
- Anwendungen und Dienste in den „täglichen Begleiter“ integrieren

J2ME für Jedermann?

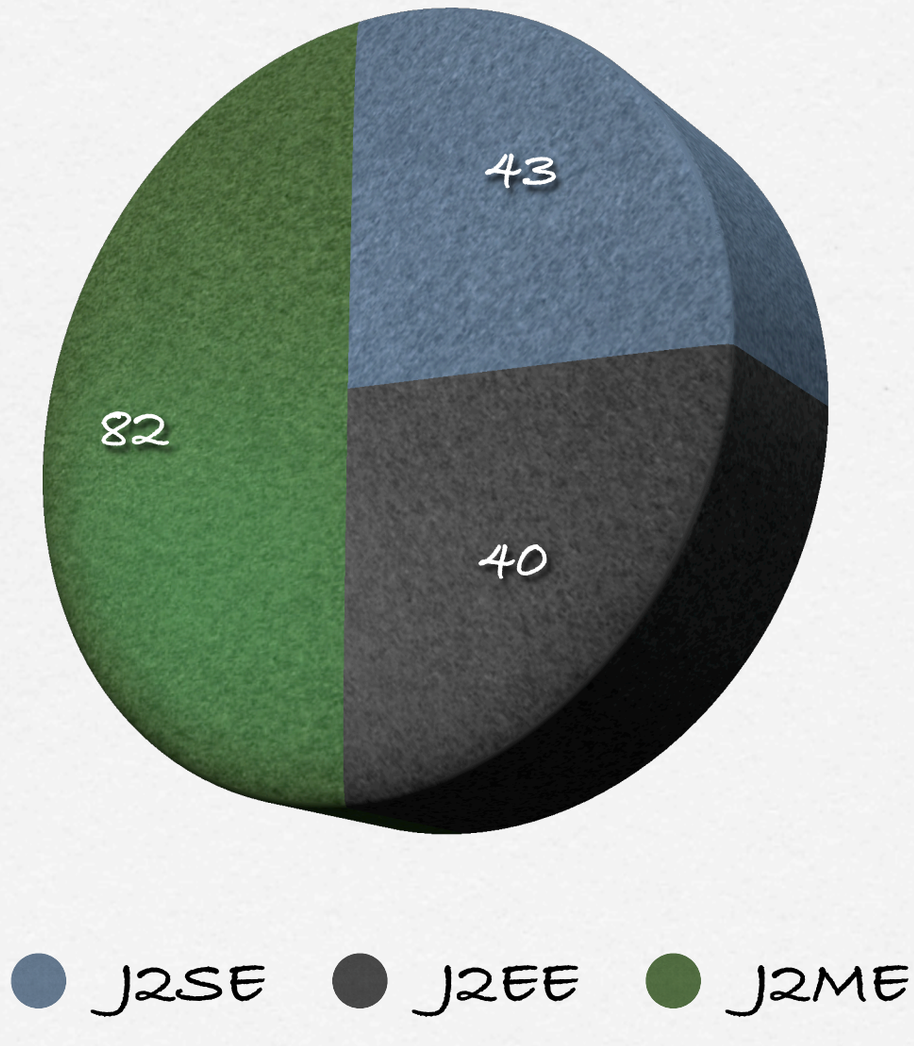
- J2ME SDKs stehen im Internet bereit
  - SUN WTK
  - Nokia, CarbideJ
  - Sony Ericsson
- J2ME Entwicklung kann „eigentlich“ jeder J2SE Softwareentwickler!

Wo ist denn
dann das
Problem... ?

Hersteller und Plattformen

- Nokia
  - S40 Ed1, Ed2 / + FP1 / FP2
  - S60 Ed1, Ed2 / FP1, FP2, FP3
  - S80 Ed1, Ed2 (obsolet mit E90)
- Motorola
  - Proprietäres OS
  - Linux
- Sony Ericsson
  - Proprietäres OS
  - JP2, JP3, JP4, JP5, JP6, JP7, JP8
- Symbian OS
  - SJP1, SJP2, SJP3
- Samsung
- LG

Java Specification Requests



27 Finale
J2ME CLDC JSRs

J2ME JSRs in Ziffern

JSR 030	JSR 139	JSR 195	JSR 232
JSR 037	JSR 172	JSR 205	JSR 234
JSR 075	JSR 177	JSR 209	JSR 238
JSR 082	JSR 179	JSR 211	JSR 248
JSR 118	JSR 180	JSR 218	JSR 256
JSR 120	JSR 184	JSR 226	JSR 257
JSR 135	JSR 185	JSR 229	JSR 271

Umbrella JSRs

Umbrella JSRs

Umbrella JSRs

JSR248

JSR185

JSR 135 (MMA 1.0)

JSR 120 (WMA 1.0)

JSR 118 (MIDP 2.0)

JSR 030 (CLDC 1.0)

Umbrella JSRs

JSR248

JSR185

- JSR 135 (MMA 1.0)
- JSR 120 (WMA 1.0)
- JSR 118 (MIDP 2.0)
- JSR 030 (CLDC 1.0)

- JSR 226 (SVG Graphics)
- JSR 205 (WMA 2.0)
- JSR 184 (3D Graphics)
- JSR 135 (MMA)
- JSR 082 (Bluetooth)
- JSR 075 (File / PIM)
- JSR 118 (MIDP 2.0)
- JSR 139 (CLDC 1.1)

Umbrella JSRs

JSR248

JSR185

- JSR 135 (MMA 1.0)
- JSR 120 (WMA 1.0)
- JSR 118 (MIDP 2.0)
- JSR 030 (CLDC 1.0)

MSA Subset:

- JSR 226 (SVG Graphics)
- JSR 205 (WMA 2.0)
- JSR 184 (3D Graphics)
- JSR 135 (MMA)
- JSR 082 (Bluetooth)
- JSR 075 (File / PIM)
- JSR 118 (MIDP 2.0)
- JSR 139 (CLDC 1.1)

Umbrella JSRs

JSR248

- JSR 238 (Internationalizat.)
- JSR 234 (MM Supplements)
- JSR 229 (Payment)
- JSR 211 (Content Handler)
- JSR 180 (SIP)
- JSR 179 (Location)
- JSR 177 (Security/Trust)
- JSR 172 (Web Services)
- JSR 226 (SVG Graphics)
- JSR 205 (WMA 2.0)
- JSR 184 (3D Graphics)
- JSR 135 (MMA)
- JSR 082 (Bluetooth)
- JSR 075 (File / PIM)
- JSR 118 (MIDP 2.0)
- JSR 139 (CLDC 1.1)

JSR185

- JSR 135 (MMA 1.0)
- JSR 120 (WMA 1.0)
- JSR 118 (MIDP 2.0)
- JSR 030 (CLDC 1.0)

MSA Subset:

- JSR 226 (SVG Graphics)
- JSR 205 (WMA 2.0)
- JSR 184 (3D Graphics)
- JSR 135 (MMA)
- JSR 082 (Bluetooth)
- JSR 075 (File / PIM)
- JSR 118 (MIDP 2.0)
- JSR 139 (CLDC 1.1)

Beispiel: Nokia 6630

- Symbian OS v8.0a
- S60 2nd Ed FP2
  - JSR 75 FConn & PIM
  - Nokia UI
  - JSR 139 CLDC 1.1
  - JSR 118 MIDP 2.0
  - JSR 82 Bluetooth (kein OBEX)
  - JSR 184 Mobile 3D Graphics
  - JSR 135 Mobile Media
  - JSR 120 Wireless Messaging

Beispiel: SE Z750

- Sony Ericsson OS
- Java Platform 8 (JP-8)
  - JSR 139 CLDC 1.1
  - JSR 118 MIDP 2.0
  - JSR 82 Bluetooth
  - JSR 120/205 WMA
  - JSR 172 Web Services
  - JSR 75 FConn & PIM
  - JSR 234 Advanced MM Suppl.
  - JSR 135 Mobile Media

Beispiel: SE Z750

- Sony Ericsson OS
- Java Platform 8 (JP-8)
  - JSR 139 CLDC 1.1
  - JSR 118 MIDP 2.0
  - JSR 82 Bluetooth
  - JSR 120/205 WMA
  - JSR 172 Web Services
  - JSR 75 FConn & PIM
  - JSR 234 Advanced MM Suppl.
  - JSR 135 Mobile Media

Beispiel: SE Z750

- Sony Ericsson OS
- Java Platform 8 (JP-8)
  - JSR 139 CLDC 1.1
  - JSR 118 MIDP 2.0
  - JSR 82 Bluetooth
  - JSR 120/205 WMA
  - JSR 172 Web Services
  - JSR 75 FConn & PIM
  - JSR 234 Advanced MM Suppl.
  - JSR 135 Mobile Media
  - JSR 185 JTWI
  - JSR 248 MSA
    - JSR 177 Security and Trust
    - JSR 179 Location
    - JSR 180 SIP
    - JSR 184 Mobile 3D Graphics 1.1
    - JSR 211 Content Handler
    - JSR 226 SV Graphics
    - JSR 229 Payment
    - JSR 238 Mobile Internationalization
  - JSR 239 OpenlGL ES API

Von Theorie zur Praxis

- [ ] UI
- [ ] Persistenz (RMS, FileConnection)
- [ ] Networking
- [ ] Over the Air (OTA)
- [ ] Sicherheit / Zertifikate
- [ ] Multi Media

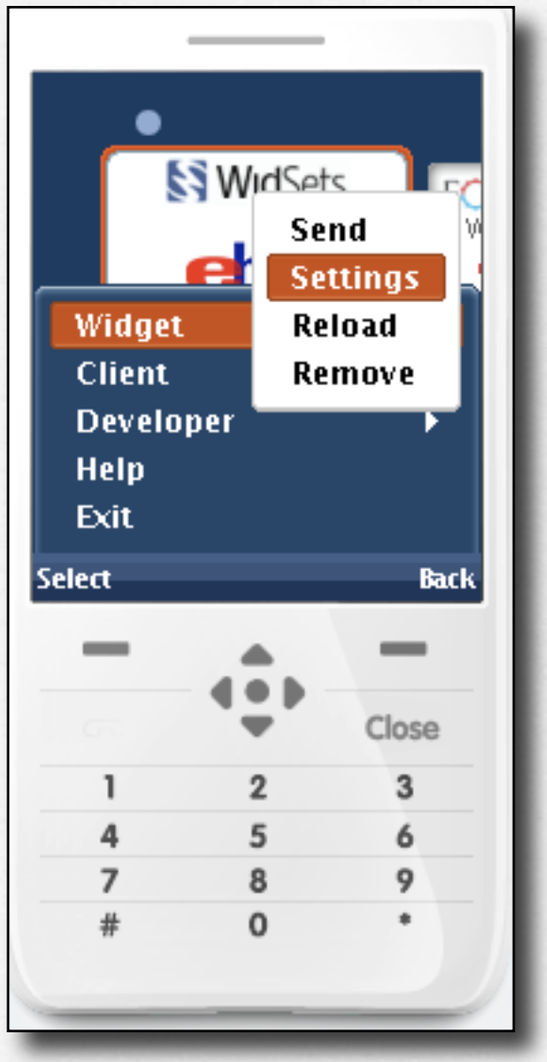
Benutzerschnittstellen (UI)

- MIDP 2.0 LCDUI
  - High Level UI -> Forms, Items, Alerts
  - Low Level UI -> Canvas
  - Canvas als Basis für eigene UIs
    - Universal Client, FonApp Inc.
    - Google Mobile Maps
    - Widsets, J2ME Widget Engine

Beispiele

Universal Client

Google Maps Mobile



WidSets

KUI

- kobjects.net Opensource Projekt
- Kommerzielle Lizenz verfügbar
- Angelehnt an LCDUI
- Kein Präprozessorcode
- Produkte die KUI einsetzen
  - qeep (Instant Messenger)
  - Joca (Community Portal)

J2ME Polish

- CSS basiertes GUI Design/Entwicklung
- Lokalisierung
- Eigenes Persistenz Framework
- RMI/RPC Framework
- Preprozessor
- Resource Handling

Beispiele: KUI / Polish

MIDlet Größe: 43kB

MIDlet Größe: 84kB

Datenpersistenz

- RMS (MIDP1.0 und MIDP 2.0)
  - JTWI: 5 unabhängige RecordStores pro MIDlet
- FileConnection (JSR 75)
  - Interner Speicher
  - Wechselmedien (MMC, RS-MMC, SD-Card, Trans Flash, Memory Stick)
  - ohne Zertifikat nicht einsetzbar!

FileConnection

- Sony Ericsson `System.getProperty()`
  - `fileconn.dir.photos`, `fileconn.dir.videos`, `fileconn.dir.graphics`, `fileconn.dir.tones`, `fileconn.dir.music`, `fileconn.dir.recordings`, `fileconn.dir.private`
- Nokia virtuelle FileSystem Roots
  - Internal, Temporary, Memory card, Rom, Images, Videos, Graphics, Sounds, Music, Recordings, Private

Netzwerk: http vs. sockets

- http Standard Protokoll der J2ME
- Aktives Server Push nur mit Sockets
- SocketConnection mittlerweile häufig unterstützt
- WICHTIG: Sockets werden im GSM Netz scheinbar nicht geschlossen ! Serverseitiges Ping notwendig.

OTA (Over-The-Air)

- „Verteilen“ einer Anwendung via SMS
- Download-URL ist in SMS codiert
- Download erfolgt mit WAP-Browser
- Geräteidentifizierung durch User-Agent
  - SonyEricssonK800i/R1ED Browser/NetFront/3.3 Profile/MIDP-2.0 Configuration/CLDC-1.1
  - Nokia6136/2.0 (p03.31) Profile/MIDP-2.0 Configuration/CLDC-1.1
- Implementierung als Servlet oder PHP

OTA in der Entwicklung

- SDKs liefern Software zur Installation
- Nokia, Sony Ericsson, Motorola, Siemens, Samsung, LG, etc.
- SDK Chaos auf dem Entwicklungs Rechner
- Bluetooth meinst ausreichend
  (Ausnahme Nokia S40)
- OTA in der Entwicklung hilfreich !

Zertifikate und Signaturen

Hersteller

Operator

CA

- Verisign
- Baltimore
- Entrust
- GlobalSign
- Thawte
- RSA Data Security

Nervige Popups

- Darf diese Anwedung eine Verbindung aufbauen?
- Darf diese Anwendung Kontaktdaten lesen?
- Seriöse Applikationen schon beim OTA Download an der Signatur erkennbar
- „Vertrauenswürdig“

Audio-/Video-Streaming

- RTSP Streaming in JSR 135 definiert
- SE W800 eines der ersten Telefone
- Fragen:
  - Freien Server im Internet verwenden?
  - Eigenen Server installieren?
  - Streams Codieren?

Aufbau Infrastruktur

- Streaming Server
  - Quicktime (Kommerziell MacOS X)
  - Darwin (Win, Linux, MacOS X)
- Encoding
  - Quicktime PRO (Aufnahmen)
  - Quicktime Broadcaster (Live)
- MIDLet
  - Manager.createPlayer("rtsp://myVideo.3gp");

Operator Proxy

- Technisch sind alle Aufgaben lösbar
- Streaming funktioniert dennoch nicht
- Tests mit verschiedensten SIM Karten
  - T-Mobile -> OK
  - Vodafone -> Nicht möglich
  - E-Plus -> Nicht möglich
  - O2 Germany -> Nicht möglich

Emulatoren

- Nokia CarbideJ 1.5 (ODD S60 3rd)
- Sony Ericsson SDK (ODD)
- Motorola Java und IDen SDKs
- Drittanbieter
  - ME4SE/XME4SE (www.kobjects.net)
  - MicroEmulator (www.microemu.org)
  - MPowerplayer (mpowerplayer.com)

Code Debugging

- Netbeans und Eclipse integrieren SDKs
- Debugging gegen SDKs möglich
- Sony Ericsson ODD, Nokia auf S60 3rd
- Präprozessoren machen Code unleserlich
- Verwendung einsparen um „debugging“ zu vereinfachen. (KUI Ansatz)

Debugging

- Bei der Masse JSRs und verfügbarer Telefone mühsamer Prozess.
- Nicht zu viele Features auf einmal implementieren.
- Häufiges testen zwischendurch
- Emulatortest ersetzt nicht OD Testing
- Verhalten abhängig von der Firmware

Zusammenfassung

- Write Once, Run Anywhere ?
- Neue JSRs meinst sehr fehlerträchtig.
- Je spezifischer die Nutzung von JSRs, desto geringer wird die Anzahl funktionierender Mobiltelefone
- Emulatoren zum Debuggen „offensichtlicher“ Fehler.
- Keine Freigabe ohne Device Tests !

Vielen Dank für Ihre Aufmerksamkeit !

Q & A ...