Fragenkatalog zur schriftlichen Prüfung, WS 2004/05

Programmieren mobiler Endgeräte 1

Prüfer: Georg Mittenecker

1. Geben Sie eine stichwort-artige Übersicht über Typen von mobilen Endgeräten und ihre wichtigsten Betriebs-Systeme!

- Tablet PC - Windows XP Tablet PC Edition
- Pocket PC - MS Windows CE / Pocket PC
  - Pocket PC
  - Pocket PC Phone Edition
  - Smartphone
- Palm / Handhelds - Palm OS
- Smartphones / Mobile D. - Symbian
- Smartphones / PDAs - Embedded Linux

2. Erklären Sie die Problematik der verschiedenen verwendeten Einheiten für Speichergrößen und Übertragungs-Raten in Informatik und Telekommunikation, geben Sie ein anschauliches Beispiel dazu an (inkl. Prozentangabe der Diskrepanz). Wie lautet die Einheit einer internationalen Norm, die das Problem lösen könnte?

Informatik:

1kByte = 1024 Byte = 8192 Bit
1MByte = 1024x1024 Byte = 1.048.576 Byte

Telekommunikation:

1kBit/s = 1000 Bit/s
1MBit/s = 1.000.000 Bit/s

Diskrepanz: ca. 2,4%

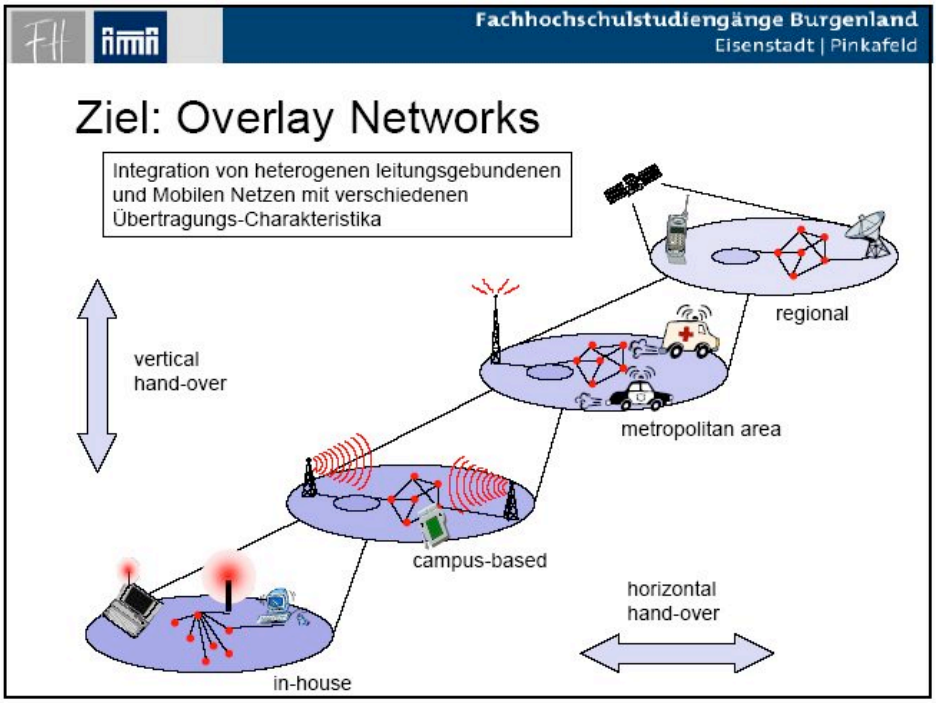
IEC-Norm: kibibyte und Mebibyte usw.
Die neuen Vorsilben sind jeweils die Abkürzung für den alten Einheitsbegriff zuzüglich der Ergänzung binary. Z.B. Kilo-binary = Kibi.

3. Geben Sie für folgende Übertragungs-Technologien eine Abschätzung der maximal möglichen Übertragungsraten an (für den Nutzer praktisch nutzbares Maximum):

Universal Serial Bus USB 1.1	**:** 12MBit/s
Universal Serial Bus USB 2.0	**:** 480 MBit/s
IEEE 1394 "Firewire"	**:** 400 MBit/s
Infrarot IrDA	**:** IrDA 1.0: 115kbit/s IrDA 1.1: 4MBit/s
Bluetooth (Version 1.1)	**:** Daten: 721 kBit/s, Sprache: 64 kBit/s
WLAN 802.11b	**:** 11 MBit/s (Folie: 5MBit/s)
WLAN 802.11g	**:** 54 MBit/s (Folie: 20 MBit/s)
GSM / GPRS	: 9,6 -14,4 kBit/s, mit HSCSD bis 57,6 kBit/s), :GPRS: 14,4 kBit/s pro Slot, bei 8 Slots max: 171,2kBit/s
UMTS	**:** bis zu 2 MBit/s (Folie: 200-384 kBit/s)

4. Erklären Sie die beiden folgenden Begriffe aus dem Bereich der mobilen Netze:

a. Vertical hand-over: zwischen unterschiedlichen drahtlosen Technologien
b. Horizontal hand-over: mit gleiche drahtlosen Technologien



5. Erklären Sie alle verwendeten Abkürzungen und geben Sie für die folgenden Java-Technologien typische Geräte-Plattformen an:

a. J2EE: Java 2 Platform Enterprise Edition (Server)
b. J2SE: Java 2 Platform Standard Edition (Workstations, PC´s, Laptops)
c. J2ME – CDC: Java 2 Platform Micro Edition (set-top box, net TV, screenphones)
d. J2ME – CLDC (Mobiltelefone, PDA´s, Pager)

6. Vergleichen Sie die „typischen Geräte“ für CDC bzw. CLDC, erklären Sie alle verwendeten Abkürzungen!

CDC: Connected Device Configuration

- für leistungsfähigere, gemeinsam genutzte, stationäre Endgeräte (z.B.: Set-Top-Boxes)
- kaum verbreitet

CLDC: Connected, Limited Device Configuration

- für persönliche, mobile Endgeräte (z.B.: Mobiltelefone) ausgelegt
- millionenfach verbreitet

7. Geben Sie einige Vor- bzw. Nachteile von J2ME im Vergleich zur Verwendung von plattformspezifischen C++-Bibliotheken an!

Vorteile:

- von allen wichtigen Geräteherstellern unterstützt
- Spezifikationen in JCP-Expertengruppen abgestimmt
- Programmierschnittstellen öffentlich zugänglich und wohl dokumentiert
- offen für OEM-spezifische Klassenbibliotheken
- höhere Abstraktion und Verzicht auf fehleranfällige Konstrukte verspricht höhere Produktivität und niedrigere Fehleranfälligkeit als C++
- kurze Einarbeitungszeit für Java-Entwickler

Nachteile:

- Nutzung der spezifischen Funktionen von Endgeräten nicht vollständig
- fehlender JIT Compiler
- niedrigere Performance (z.B.: Video Codecs)

8. Vergleichen Sie die prinzipiellen Eigenschaften von J2ME Konfigurationen (CDC bzw. CLDC) und Profilen (MIDP 1.0 oder 2.0), erklären Sie alle verwendeten Abkürzungen!

JSR	Gegenstand	Ver.
30	Connected, Limited Device Configuration (CLDC)	1.0
139	Connected, Limited Device Configuration (CLDC)	1.1
36	Connected Device Configuration (CDC)	1.0

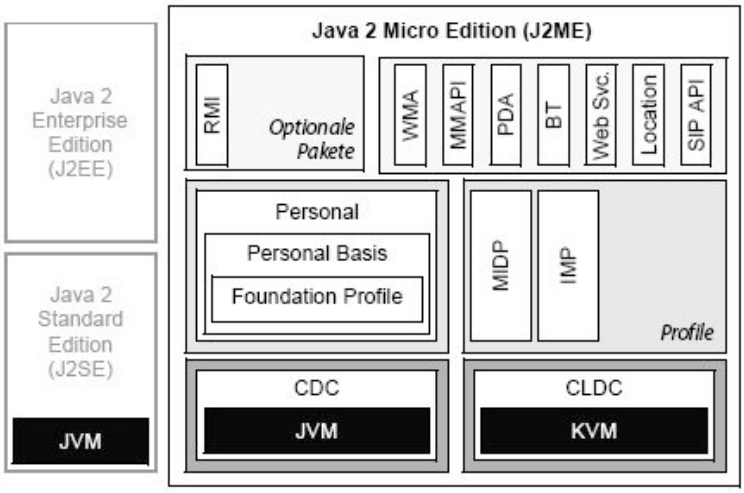
J2ME-Konfigurationen

JSR	Gegenstand	Ver.	CLDC	CDC
37	Mobile Information Device Profile (MIDP)	1.0	✓	
118	Mobile Information Device Profile (MIDP)	2.0	✓	
195	Information Module Profile (IMP)	1.0	✓	
228	IMP – Next Generation	2.0	✓	
46	Foundation Profile	1.0		✓
129	Personal Basis Profile	1.0		✓
62	Personal Profile	1.0		✓

J2ME-Profile

Überblick:

Editionen, Konfigurationen, Profile und optionale Pakete



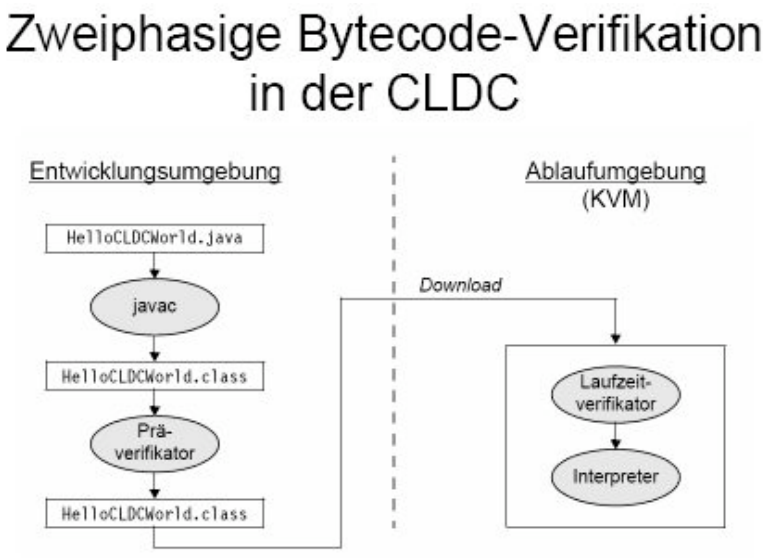
9. Geben Sie die volle Bedeutung der folgenden Abkürzungen/Begriffe (aus dem Bereich der Java-Programmierung mobiler Geräte) an:

a. J2ME Java 2 Micro Edition
b. JSR Java Specification Request
c. JTWI Java Technologie fort the Wireless Industry
d. JVM Java Virtual Machine
e. KVM Kilobyte Virtual Machine
f. JAR Java Archive
g. JAD Java Apllication Descriptor
h. AMS Application Management Software
i. WTK Wireless Toolkit
j. LCDUI Limited Connected Device User Interface
Lowest Common Denominator Interface
Liquid Crystal Device User Interface
k. RMS Record Management System

10. Erklären Sie stichwortartig (ev. mit Ablauf-Diagramm) die zweiphasige Bytecode-Verifikation bei der CLDC.

1. Der *Präverifikator* verarbeitet zunächst die vom Compiler generierte *.class*-Datei. Der Präverifikator hat die Aufgabe, zeit-und ressourcenintensive Überprüfungen vorzunehmen und
eine semantisch äquivalente *.class*-Datei zu erzeugen, die sich zur Laufzeit effizient überprüfen lässt. Üblicherweise läuft der Präverifikator in der Entwicklungsumgebung, in der der Quellcode
geschrieben und übersetzt wird.

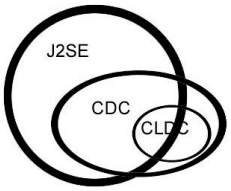
2. Der *Laufzeitverifikator* erhält eine präverifizierte *.class*-Datei und überprüft die darin enthaltenen Bytecodes, bevor sie im Interpreter zur Ausführung gelangen.



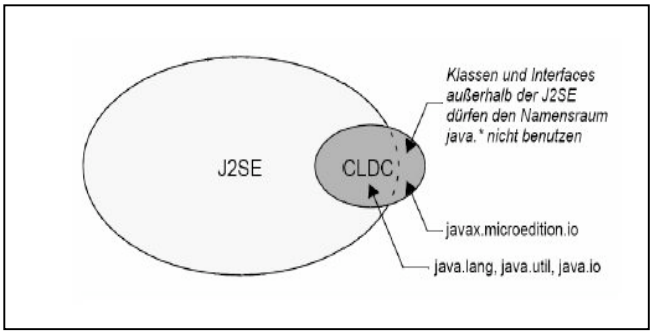
11. Geben Sie an, aus welchen drei J2SE-Klassenbibliotheken Klassen und Interfaces von der J2ME übernommen wurden.

java.lang, java.util und java.io

12. Ist die CLDC-Klassenbibliothek eine echte Teilmenge der J2SE-Bibliothek (kurze Erklärung!)?



Es ist auf der nebenstehenden Figur klar ersichtlich, dass die CDC und CLDC nicht ausschliesslich aus J2ME Klassen bestehen müssen, sondern dass die jeweilige Applikation um eigene Klassen erweitert werden kann. So gibt es standardmässig keine JDBC Konnektivität bei J2ME, es existiert aber ein Produkt eines Drittanbieters, welches die beschränkte Anbindung direkt an Datenbanken via JDBC ermöglicht.

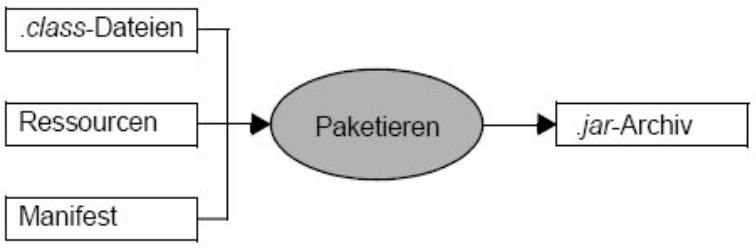


13. Erklären Sie die Begriffe „MIDlet“, „MIDlet Suite“, Manifest, jad und jar!

MIDlet: Anwendungen für das MIDP heißen *MIDlets*. Ein MIDlet ist eine gewöhnliche Java-Klasse, die von der abstrakten Klasse MIDlet aus der MIDP-Bibliothek abgeleitet ist und Implementierungen für die Lebenszyklusmethoden zum Starten, Pausieren und Beenden enthält. Daneben kann die Anwendung weitere benutzerdefinierte Klassen und Interfaces umfassen.
MIDlet-Suite: Ein oder mehrere MIDlets bilden eine *MIDlet-Suite*. Eine MIDlet-Suite ist die kleinste installierbare Einheit; isolierte MIDlets sind für den Zweck der Installation ungeeignet. Die Zugehörigkeit eines MIDlet zu einer MIDlet-Suite hat verschiedene Auswirkungen, und zwar unter anderem die folgenden:

Eine MIDlet-Suite wird als ***.jar-Archiv*** verpackt, dessen Inhalt sich zusammensetzt aus:

- den präverifizierten *.class*-Dateien der enthaltenen MIDlets,
- ggf. weiteren *Ressourcen* wie Icons, Grafiken, Hilfetexten usw. und
- einem *Manifest*.



Das ***Manifest*** ist eine Textdatei mit dem Namen *MANIFEST.MF*, die Metainformationen in Form von Attributvereinbarungen enthält.

Für den Applikationsdeskriptor, eine Datei mit der Endung ***.jad***, gilt dieselbe Syntax wie für das Manifest und auch bei den vorgesehen Attributen gibt es weitgehende Übereinstimmung. Der Mehrwert des Applikationsdeskriptors wird beim Installieren der MIDlet-Suite deutlich: Mit der relativ kompakten *.jad*-Datei existiert ein Mechanismus, wichtige Informationen über eine MIDlet-Suite in der Zielumgebung bekannt zu machen – noch bevor das umfangreichere *.jar*-Archiv übertragen wird.

Welches Konzept für die Bereitstellung von Software für Mobilgeräte steckt dahinter? Welche Inhalte werden damit jeweils transportiert?

14. Welche Möglichkeiten des Zugriffs über gemeinsam genutzte Variablen oder permanent gespeicherte Daten („persistente Datenbestände“) gibt es im Zusammenhang mit mehreren MIDlets und einer bzw. mehrerer MIDlet Suiten?

Persistente Datenbestände, die von einem MIDlet angelegt werden, sind für die anderen MIDlets sichtbar und zum Lesen, Überschreiben und Löschen freigegeben. Dagegen lässt sich der Zugriff für MIDlets aus anderen MIDlet-Suites sperren oder auf lesende Operationen beschränken.

15. Was ist die Application Management Software (AMS)? Welche Aufgaben hat sie?

Die AMS ist eine essenzielle Komponente in jedem MIDP-kompatiblen Gerät. In ihren Verantwortungsbereich fällt auch die Steuerung des MIDlet-Lebenszyklus, zu dem Aktionen wie das Starten, Pausieren und Beenden gehören.

16. Geben Sie den „Programm-Rumpf“ eines J2ME MIDlets an, das alle obligatorisch zu implementierenden Methoden enthält!

```
package xy;

import javax.microedition.midlet.MIDlet;
```

```
public class HelloMIDPWorld extends MIDlet
{
  protected void startApp()  {
        // System.out.println("Hello, MIDP World");
  }

  protected void pauseApp () {
  }

  protected void destroyApp (boolean unconditional) {
        // System.out.println("Goodbye, Cruel World");
  }

}
```

17. Beschreiben Sie stichwortartig (möglichst mit Skizze) den Lebenszyklus eines MIDlets (drei Haupt-Zustände und Übergänge dazwischen). Geben Sie auch an, wie/woher ein Zustands-Übergang ausgelöst werden kann!

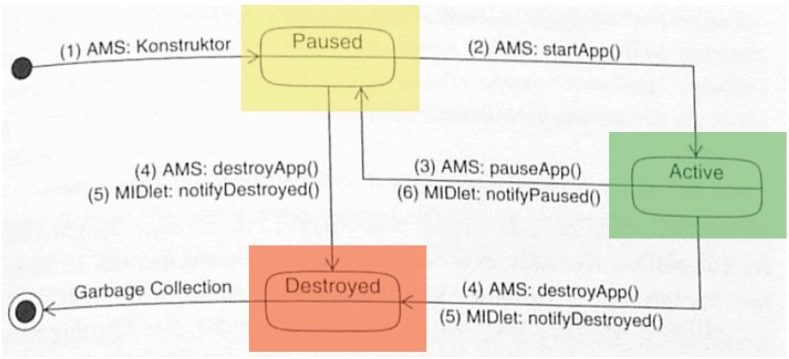
MIDlet – Zustände

- **Paused**
  - **Initialisiert, Display-Ausgaben unmöglich**
  - **Möglichst keine NW-Verbindungen, große Objekte und Threads aufrechthalten**

- **Active**
  - **„im Vordergrund"**
  - **Display-Ausgaben und Nutzereingabe möglich**

- **Destroyed**
  - **Beendet; keine Ressource mehr; Threads terminiert; NW-Verbindungen geschlossen; Daten gespeichert**
  -

MIDlet Lebenszyklus



18. Erklären Sie die Eigenschaften der Methode platformRequest() und die damit verbundenen Möglichkeiten!

Eigenschaften:

- nur in MIDP 2.0 vorhanden
- Starten externer Anwendungen durch URL
- vorgeschrieben:
  - URL auf .jad oder .jar → Installation der Suite
  - URL tel:<nummer> führt Anrufe durch
- optional weitere Schemata, z.B.:
  - rtsp://host/clip.mp4 startet Video-Player

Möglichkeiten:

bei genügend Ressourcen:

- ausgewählte Applikation im „Vordergrund" starten
- **platformRequest()** liefert sofort **false**, ohne auf Starten der Applikation zu warten
- MIDlet bleibt im „Hintergrund" erhalten

bei ungenügenden Ressourcen:

- **platformRequest()** liefert **true**, Start des (letzten) platformRequest()-Aufrufes erfolgt nach Beenden des MIDlets.

19. Erklären Sie die Unterschiede zwischen J2SE und J2ME bei der Verwendung von System.in, System.out und System.err!

- Klasse System in CLDC und MIDP besitzt kein System.in
- System.out bzw. System.err funktioniert nur auf Emulator, nicht auf realen Endgeräten

20. Welches Paket bzw. welche prinzipiellen Möglichkeiten davon sind teilweise durch die Sicherheitsmechanismen von J2ME eingeschränkt?
Was bedeuten die Begriffe „blanket", „session" bzw. „oneshot" im Zusammenhang mit User Permissions?

blanket: bis zur Deinstallation der MIDlet-Suite
session: während eines Programmablaufes gültig
oneshot: nur für aktuellen Aufruf

21. High Level LCDUI - Kurze Beispiele mit: Display, List, Alert, TextBox, Form, Command (Erklären eines Beispieles, soweit in Vorlesungsunterlagen erklärt!)

List – zur Auswahl von Einträgen aus vorgegebener Menge
TextBox – für Texteingaben
Alert – für Anzeigen von Rückmeldungen
Form – für komplex strukturierte Dialoge

- **nicht vollständig → siehe Folien Bedienoberflächen**
-

22. Low Level LCDUI - Kurze Beispiele mit: Canvas, CustomItem.. (Erklären eines Beispieles, soweit in Vorlesungsunterlagen erklärt!)

Beispiel für `Canvas`

```
public class HelloCanvas extends Canvas {
   private static final String TEXT = "HelloCanvas";

   protected void paint(Graphics g) {
      int w = getWidth();        // Breite des Canvas-Inhalts
      int h = getHeight();       // Höhe des Canvas-Inhalts
      g.setColor(0xc08040);      // 0xRRGGBB (rot, grün, blau)
      g.fillRect(0, 0, w, h);    // Ausgefülltes Rechteck
      g.setColor(0x000000);      // Schwarz
      g.drawString(TEXT, w / 2, 30, Graphics.TOP|Graphics.HCENTER);
   }
}
```

Beispiel für `CustomItem`

```
public class HelloItem extends CustomItem {
   private static final String TEXT = "HelloItem";

   protected void paint(Graphics g, int w, int h) {
      g.setColor(0xc08040);      // 0xRRGGBB (rot, grün, blau)
      g.fillRect(0, 0, w, h);    // Ausgefülltes Rechteck
      g.setColor(0x000000);      // Weiß
      g.drawString(TEXT, w / 2, 30, Graphics.TOP|Graphics.HCENTER);
   }

   // ... Implementierungen von weiteren abstrakten Methoden ...
}
```

23. GCF - Kurze Beispiele mit: Connector (Erklären eines Beispieles, soweit in Vorlesungsunterlagen erklärt!)

Beispiel für `Connector.open` einer HTTPConnection

```
HttpConnection con = null;
try {
   con = (HttpConnection)Connector.open("http://www.dpunkt.de/");
   …
} catch (ConnectionNotFoundException e) {
   …  // Verbindung mit Server kann nicht hergestellt werden
} catch (IOException e) {
   …  // Allgemeiner Ein/Ausgabe-Fehler
} catch (SecurityExeception e) {          // Unchecked Exception
   …  // Keine Berechtigung für Netzwerkzugriffe
} finally {
   // Connection im Erfolgs- und Fehlerfall ordentlich schließen
   if (con != null) {
      try { con.close(); } catch (IOException e) {}
   }
}
```

24. Wozu dient die „Push Registry“? Welche Möglichkeiten zur Aktivierung gibt es dabei? Welche Software-Komponente kann dabei ein MIDlet aktivieren?

Die Push-Registry ist eine Neuerung in MIDP 2.0 und steuert das Starten von MIDlets, was bisher nur dem Benutzer möglich war, anhand bestimmter Ereignissen.

***Der Rest dieser seltsamen Fragestellung ist mir unbegreiflich**

25. Was bedeutet RMS? Welche Funktionen bietet das RMS? Wozu benötigt man eindeutige IDs (Primärschlüssel)? Kann auf einen Datensatz auch ohne ID zugegriffen werden?

RMS – Record Management System
Ist eine einfache, satzorientierte Datenbankschnittstelle, über die sich persistente Daten im Endgerät verwalten lassen

Funktionen:
- Dauerhafte Speicherung von Daten
- Benutzerpräferenzen
- Arbeitsergebnisse
- Dokumente
- Nur 8 kB Minimum vorgeschrieben
- Muss auch nach Ausschalten und Batteriewechsel erhalten bleiben

Zugriff auf Datensatz ohne ID durch indexsequenziellen Zugriff

26. Was bedeutet MMAPI? Erklären Sie das Konzept der MMAPI (mit dem Zusammenspiel der wichtigsten Komponenten), verwenden Sie dazu eventuell eine Skizze.

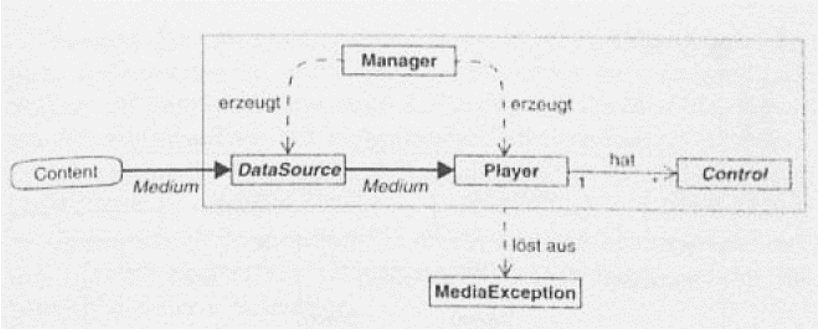
Mobile Media API

- Polyphone Klingeltöne
- Integrierte Digitalkamera
  - Fotos oder auch Videos aufnehmen möglich
- JSR-135 -> MMAPI 1.1
- Nicht an bestimmte Profile oder UI-Bibliothek gebunden
- Kann als „leichtgewichtige Variante des Java Media Framework" verstanden werden

MMAPI Konzepte

- Allgemeine **Programmierschnittstelle für Verarbeitung von zeitabhängigen Medien:**
  - Töne, Musikstücke, Filme
- Sehr **flexibel für unterschiedliche Quellen** (Datei, Server, Kamera..) und Dekodierung verschiedener Formate
  - Aber Implementierung kann nur bestimmte Codecs anbieten

Zusammenspiel der wichtigsten Komponenten



27. Welche Vor- bzw. Nachteile hat die Verwendung von optionalen Paketen bei der J2ME-Programmierung?

- Verwendung von optionalen Paketen **schöpft die Möglichkeiten der Ablaufumgebung eventuell stärker aus**
- Damit aber die **Portabilität eingeschränkt**

28. Welche Art von Nachrichten kann man mit der Wireless Messaging API (1.1 bzw. 2.0) senden bzw. empfangen?

WMA unterstützt das Versenden und Empfangen von Mitteilungen wie SMS und ab v.2.0 MMS

29. **Welche Arten von „persönlichen Daten“ können mit dem PIM (Personal Information Management) Optional Package verwaltet werden?**

    - **Kontakte (Contact)**
    - **Aufgaben (ToDo)**
    - **Termine (Event)**

30. **Was sind Vor- bzw. Nachteile bei Verwendung des „FileConnection Optional Package“ im Vergleich zum RMS?**

    Vorteil von RMS: Sortier und Filtermechanismen werden unterstützt
    Nachteil von RMS: Nutzung ist Java-Programmen vorbehalten

    Vorteil von FileConnection-API: Austausch von Informationen zwischen MIDlets und nativen Applikationen ist möglich
    Nachteil von FileConnection-API: macht Dateien nur als sequenziell bearbeitbare Byte-Sequenz zugänglich

31. **Welche beiden unterschiedlichen Arten der Lokalisierung werden prinzipiell von der „Location API“ unterstützt?**

    - Bestimmung des aktuellen Ortes
    - Bewegungsgeschwindigkeit

Grün – zuverlässig richtig
Rot – muß nachgefragt werden
Blau - Hinweis