Java Video and Audio in Consumer Devices:
JMF and MM API

By

SHAHZAD NASEEM

Outline

Introduction (1/2)

Audio and Video in Consumer Devices, two alternatives:

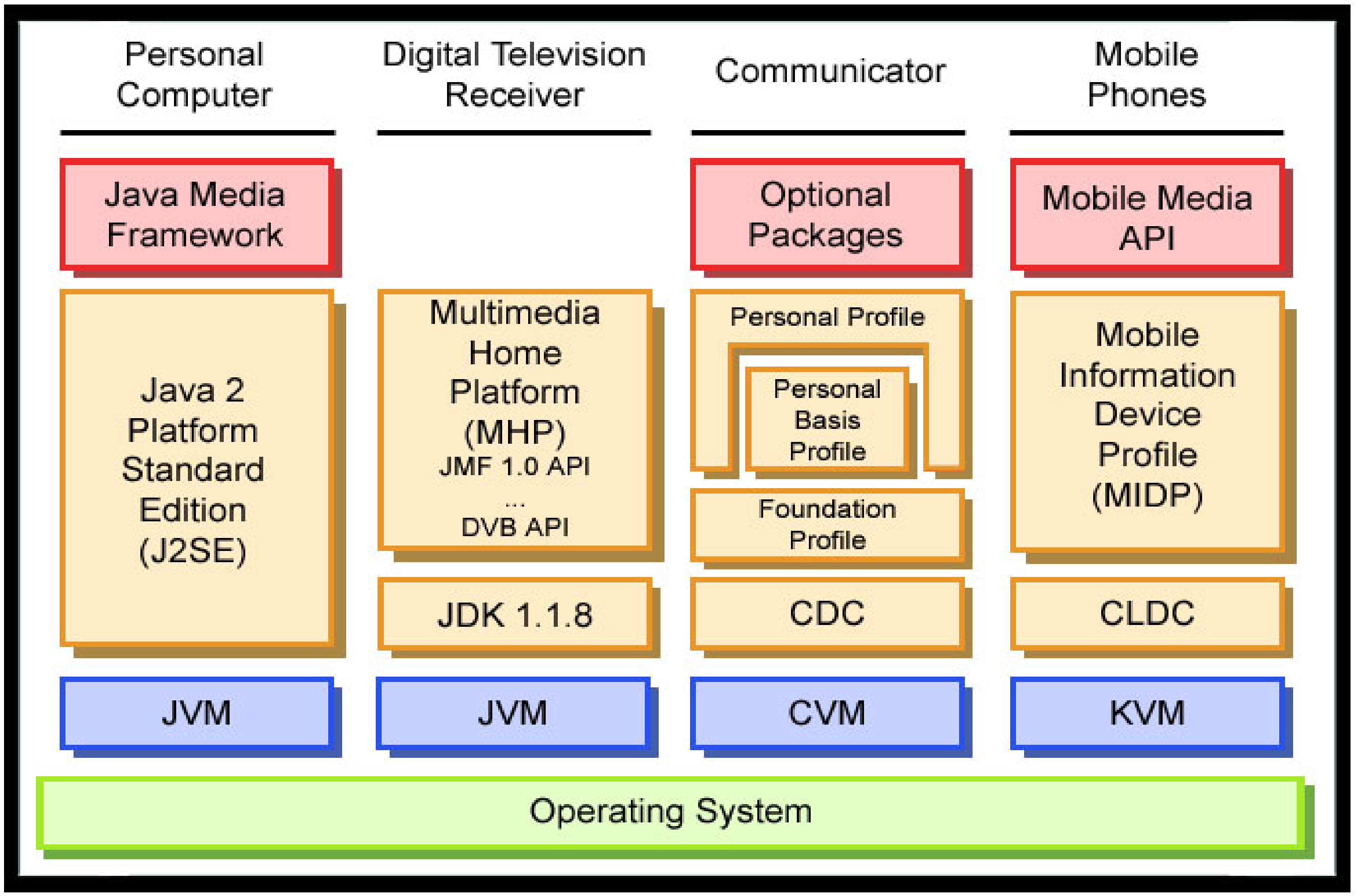
- Video objects in programming tools:
  - Synchronized Markup Integration Language (SMIL)
  - Flash presentations
  - Java Media Framework (JMF)

- Stand alone video player:
  - Proprietary players: Real One, Windows Media Player...
  - Open Source players: winamp, MPlayer...

Introduction (2/2)

- Specifically Java, why?
  - Consumer devices includes Java
    - Digital TV = Multimedia Home Platform (MHP) and JDK
    - PCs = Java 2 Standard Edition
    - Mobile phones = Connected Limited Device Configuration (CLDC) + Information Device Profile (MIDP)

Overview


Personal Computer
Digital Television Receiver
Communicator
Mobile Phones
Java Media Framework
Optional Packages
Mobile Media API
Java 2 Platform Standard Edition (J2SE)
Multimedia Home Platform (MHP)
JMF 1.0 API
...
DVB API
Personal Profile
Personal Basis Profile
Foundation Profile
Mobile Information Device Profile (MIDP)
JDK 1.1.8
CDC
CLDC
JVM
JVM
CVM
KVM
Operating System

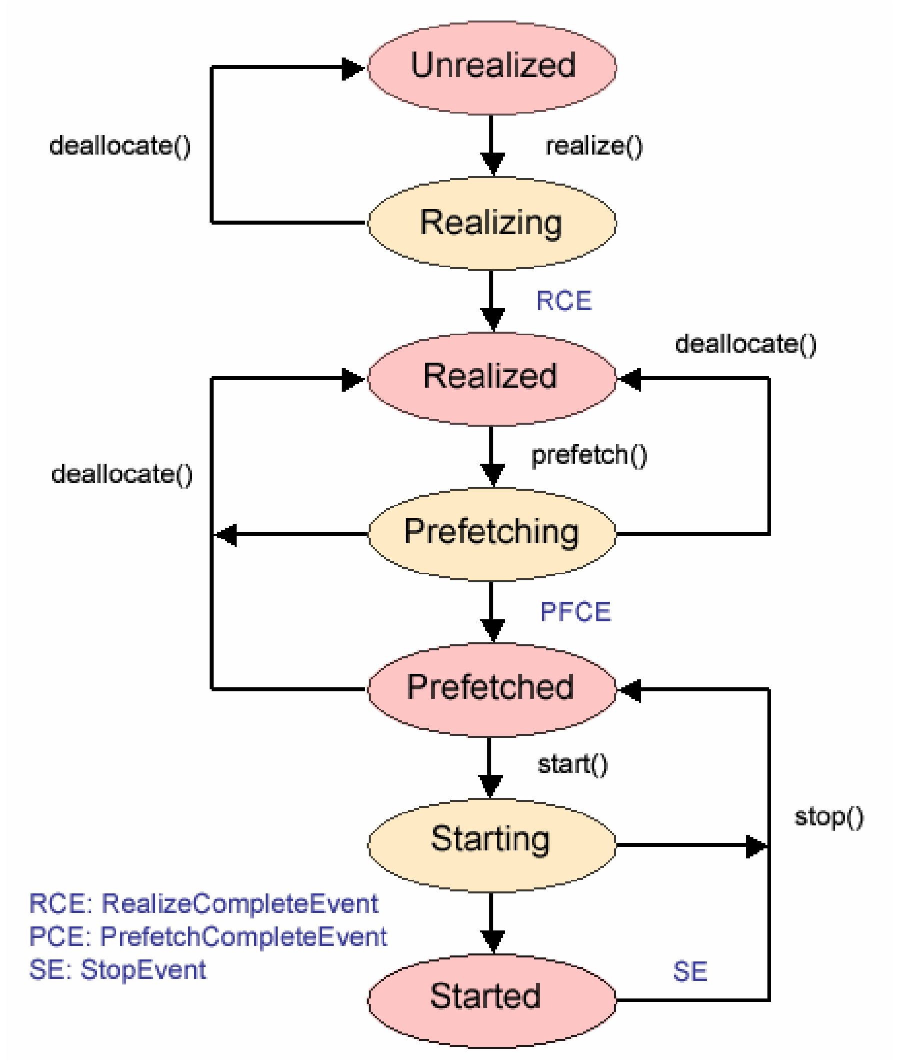
Personal Computer (1/4)

- **Physical Characteristics:**
  - Pointer device (e.g., mouse) and keyboard as major input mechanisms
  - Screen resolution: 640x480 to 1600x1200 pixels
  - Runtime memory: around 128 to 256 MB

Personal Computer (2/4)

- JMF as an optional package either version 1.0 or 2.0
- JMF relies on the mentioned classes: Manager, Player, Data Source, and Controller.
- Player behaviour:
  - Unrealised
  - Realised
  - Prefetched
  - Started

Personal Computer (3/4)



RCE: RealizeCompleteEvent
PCE: PrefetchCompleteEvent
SE: StopEvent

Personal Computer (4/4)

- Time model
- Data model: DataSource encapsulates both the location and protocol of media
- JMF 2.0 includes as well a low level API
  - A Processor performs user-defined processing on the media data using JMF plug-ins (e.g. *Codec*, *Demultiplexer*, *Effects*, *Multiplexer*, *Renderer)*
- Render is done in an AWT Component

Digital Television Receiver (1/2)

- Multimedia platform in the living room
- Services:
  - Audio visual stream (normal passive watching)
  - Interactive services (active behaviour)
- Physical Characteristics
  - Remote control as major input mechanism
  - Screen resolution: 720x576 pixels (minimal)
  - Runtime memory: at least 16 MB

Digital Television Receiver (2/2)

- Includes a number of specific digital TV controls:
  - Media Select: changes the actual stream presented (e.g., change the angle of a camera)
  - Language: intended to control the audio and subtitles (if present) language

Mobile Phone (1/5)

- Provide a number of services:
  - Internet (e.g., Nokia + Opera), MMS, Video Player
  - Office capabilities
- Physical characteristics
  - Key Pad as major input mechanism
  - Screen resolution: 84x48 to 120x130 pixels
  - Runtime memory: 160 to 512 KB
- Java Environment
  - Mobile Information Device Profile (MIDP) version 1.0 or 2.0

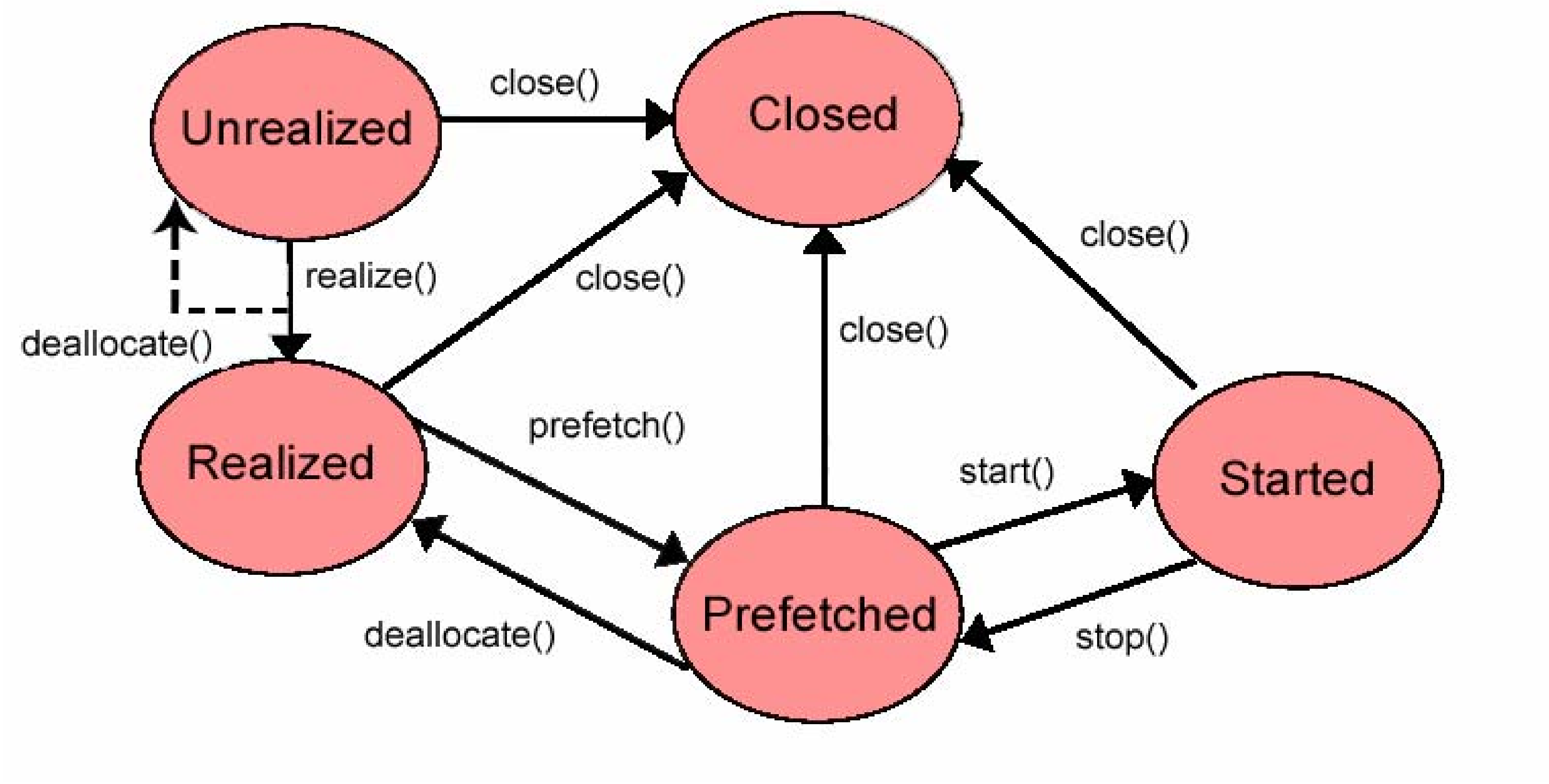
Mobile Phone (2/5)

- MMAPI Description:
  - It extends MIDP functionality by providing audio, video and other time-based multimedia support
  - It is a thin Java layer completely platform dependent
- MIDP 2.0 includes the audio-only subset from MMAPI (i.e. Audio Building Block)

Mobile Phone (3/5)

- Same concepts as JMF: Player, Controller, Manager and DataSource.
- Player behaviour same (Unrealised, Realised, Prefetched, Started) as JMF but one more state:
  - Closed: the player cannot be used again, it has released most of the resources

Mobile Phone (4/5)



Mobile Phone (5/5)

- The types of media supported depends on the *controls* associated to the Player
  - *Player.getControls()* returns the supported controls

  A Player renders media data in a component dependent on the device configuration, two options:
  - AWT Component
  - MIDP Canvas or Items
- VideoControl manages the location and the size of the video

Conclusions (1/2)

- The actual capabilities of the targeted device is the cause of the differences between standards
- Low level versus high level control of the media:
  - In MM API and JMF (MHP) the actual control of the media is done at the native level since they are resource-constrained devices (e.g. decoding)
  - JMF uses two profiles, JMF plug-ins enables developers to process the data (e.g. multiplexing)
- Behaviour of the player
  - All the standards have the same player behaviour. But MM API defines a Closed state to make explicit that all the resources are freed

Conclusions (2/2)

- MM API is influenced by the design of JMF, and have number of similarities: Manager, Player, Data Source, and Controller concepts.
  - MM API hides Controller within Player
- Video render
  - JMF = AWT Component (complete integration)
  - MHP = A layer (transparency can be applied)
  - MM API = MIDP canvas (minimal integration)
- Controls are different depending on the targeted device (e.g., subtitles language in television)
- MHP includes specific television requirements (e.g., Clock not needed in broadcast, Locators)

References

- Multimedia

  http://www.tml.hut.fi/Opinnot/T-111.350/index_uk.html
- Digital Television

  http://www.mhp-interactive.org
- PC

  http://java.sun.com/products/java-media/jmf/index.html
- Mobile Phones

  http://java.sun.com/products/j2me/

  http://java.sun.com/products/cldc/

  http://wireless.java.sun.com/midp/

  http://java.sun.com/products/mmapi/