

# Getting Started with Java Development – Best Practices

Ravi Belwal  
Sr. Technology Consultant



# Best Practices to sweeten your MIDlet

# Motivation

- Aim to better performance in Java ME
- Fragmentation
- Nokia S60 Java highly committed to performance

# Outline

- Essentials
- Graphics Operations
- Resource Usage
- Other tricks
- Usability
- Perceived Performance

## Java ME Essentials (1/2)

- System threads: do as few as possible
- Long running jobs in separate threads
- Minimize resource sizes
- Dereference objects

## Java ME Essentials (2/2)

- Catch Runtime Exceptions and Errors
- Object re-usage
- Use standard APIs
- Use obfuscator to minimize size of classes

## Demo: String vs. StringBuffer

```
String myString = new String ("Connecting ");  
myString += "people!";
```

**VS.**

```
StringBuffer myString = new StringBuffer("Connecting ");  
myString.append("people!");
```

## Demo: File Connection reuse

```
FileConnection fconn = null;
for(int i=0; i<folderCount; i++) {
    fconn = (FileConnection) Connector.open(newUrl[i]);
    doSomething(fconn);
    fconn.close();
}
```

**VS.**

```
FileConnection fconn = (FileConnection) Connector.open(newUrl[0]);
for(int i=1; i<folderCount; i++) {
    doSomething(fconn);
    fconn.setFileConnection(newUrl[i]);
}
fconn.close();
```

## LCDUI Graphics (1/3)

- Graphics drawing is always costly
- Do not create many Canvases
- Double buffering
- Avoid synchronization in paint method

## LCDUI Graphics (2/3)

- Combine images



- Be smart on `hideNotify()`
- Clear graphics is important
- Handle `sizeChanged()`



## LCDUI Graphics (3/3)

- Clip area
- Avoid over-running CPU with rendering
- Dispose used resources

# Demo: Image loading

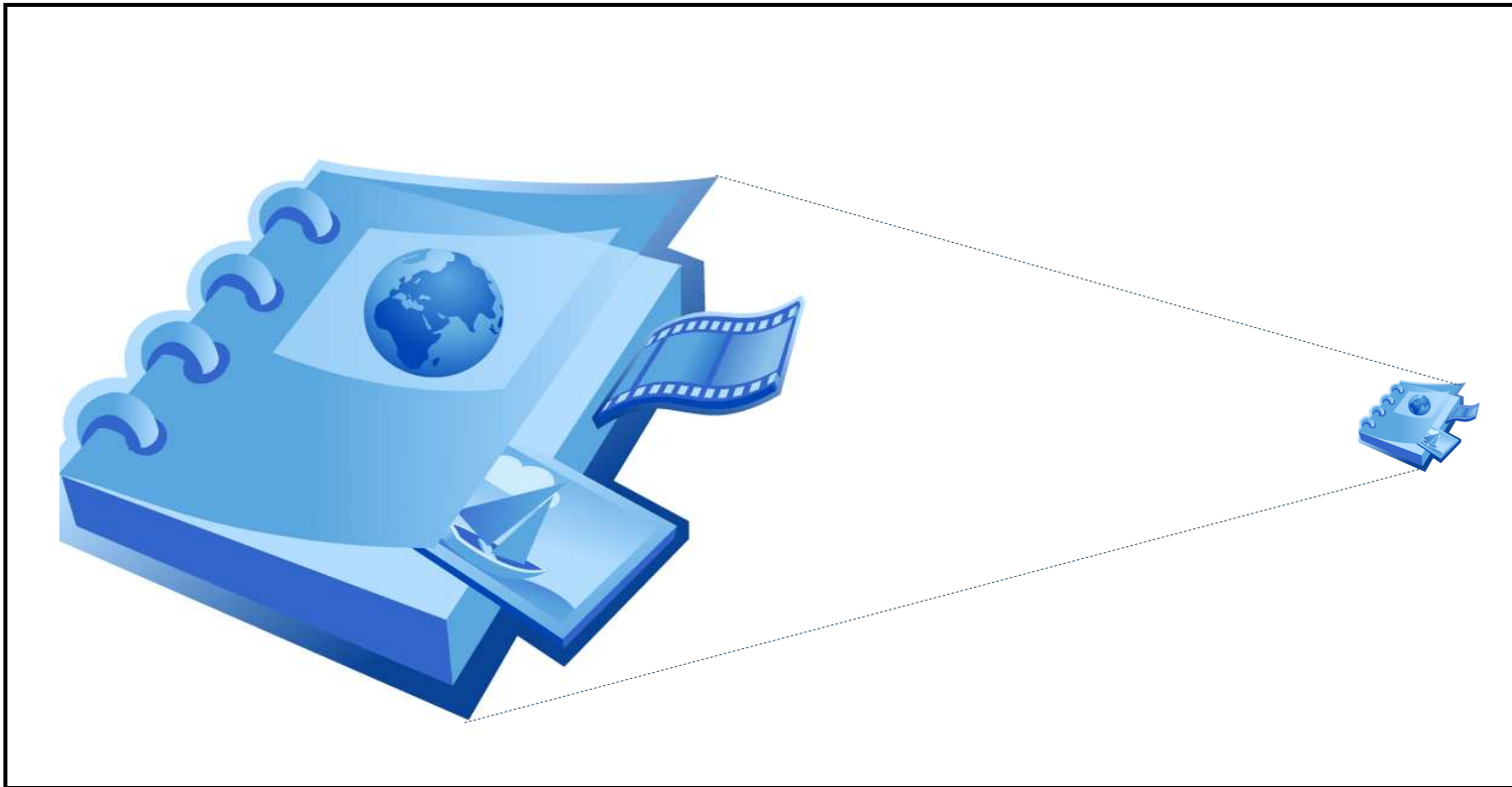


VS.



## SVG Graphics (1/2)

- Details in the SVG image



## SVG Graphics (2/2)

- Check the SVG authoring tool generated files
- Be careful about the frame rate
- Pre-rendering



# Demo: SVG strip down



**31 KB**

**VS.**



**19 KB**

## 3D Graphics

- <http://forum.nokia.com>  
Mobile Games With The M3G API
- <http://wiki.forum.nokia.com>  
TSJ000561 - M3G performance hints for 3D hardware-accelerated devices

# Graphics 2D/3D

- Do not mix 2D and 3D graphics for HW accelerated devices



2D  
background

3D  
background

2D  
spectators

3D field

2D players

2D HUD &  
fonts

## Resource Usage: RMS

- Do not open and close frequently
- Do not read and write one or two bytes
- RMS store size limitations

# Multimedia

- Optimal multimedia formats
  - `getSupportedContentTypes()`
  - `getSupportedProtocols()`
- Call de-alloc on a non used player
- Avoid reloading media items from network
- For less delay use pre-fetched player

## Network usage

- Use an asynchronous messaging model
- `HttpConnection.getLength()`
- Data compression
- Use client cache if possible

## Others: Power consumption

- Profile the power consumption of your MIDlet
  - Nokia Energy Profiler



- Use less power consuming algorithms
- Pause the MIDlet when `hideNotify()` is called

## Other tricks

- If possible avoid
  - data conversion
  - floating point and double calculation
  - too much optimization for one device
- Check MIDlet performance on different devices
- Throwing/catching exceptions are expensive
- Garbage collection

# Usability

- Smooth running of application
- Users do not remember any data across screens
- Avoid horizontal and vertical scrolling
- One hand or two hands interaction



# Perceived Performance

- Feels faster execution
- No pauses in application flow
- Inform end user about application functioning
- Device look and feel

**There is no silver bullet**  
**Pay attention to design**

# Discussion