Developing Games for J2ME™ Technology-Enabled Mobile Devices

Nicolas Lorain
Sr. Product Manager
Wireless Java™ Technologies
Sun Microsystems, Inc.
Session Speakers

- Nicolas Lorain—Sr. Product Manager, Wireless Java Technologies, Sun Microsystems
- Dave Lloyd—Chief Technical Officer, nGame
- Thomas Landspurg—Chief Technical Officer, In-Fusio
- Ariel Levin—Staff Engineer, Sun Microsystems
Explore technical issues related to the development of game applications for mobile devices, and how to address them with Java™ 2 Platform, Micro Edition (J2ME™) technology.
Objectives

- Identify key issues related to game development on mobile devices
- Learn how the J2ME platform can be used to address these issues
- Review recommended development practices
- Analyze sample MIDP-based applications
About Nicolas Lorain

- Senior Product Manager at Sun Microsystems, in charge of CLDC, MIDP, and the J2ME Wireless Toolkit
- Moderates Sun Microsystems’s KVM-INTEREST mailing list
- Speaks frequently on wireless and J2ME technologies at a number of industry events (e.g., 3GSM World Congress, PalmSource)
A Few Facts

- There will be 100 million mobile game players by 2006.
  Strategy Analytics, May 2001

- J2ME VM sales will reach over 680 million units in 2004.
  ARC group, July 2001

- There are now over 50 J2ME platform-enabled handset models available from 15 manufacturers: Casio, Fujitsu, Hitachi, LG, Mitsubishi, Motorola, NEC, Panasonic, RIM, Samsung, Sharp, Siemens, Sony and Toshiba.
nGame Ltd, In-Fusio and Sun Microsystems will address the following items:

- Platform Differences
- Portability
- Business Model
- Future Directions
Dave Lloyd
Chief Technical Officer
nGame
Managing Director and founder of Oxford and Cambridge Compilers Ltd where he designed and built compilers for scientific and engineering applications

nGame Chief Technical Officer and co-founder

Two years experience building the architecture for nGame’s innovative Content Server and Game Scripting Language
Challenge 1: Limited Memory

- Practical limit of JAR-file about 100kb
- Typical memory limit about 200kb
- Code and data trade off in memory usage

Possible tactics:
- Avoid many classes as class overhead is several kb
- Use the default package
- Obfuscate the code to reduce length of identifiers
- Work with a server to fetch assets on the fly
Challenge 2: Screen Estate

• Many different screen sizes ranging from 90x90 to 300x300
• Monochrome, greyscale or colour? Which colours?
• No AWT or similar in MIDP

Possible Tactics:
  µ Images should be available in variety of resolutions and colour formats
  µ Target a small number of typical layouts and allow ‘stretch’
  µ Implement simple AWT clone with layout manager
  µ Use Forms wherever possible
Challenge 3: User Input

- Keypad is common but not guaranteed
- Keypad layout and usability is very variable—important for playability
- Touch screens on higher end devices

Possible Tactics:
- Separate Keypad and Touch Screen UI designs
- Include a button area on Touch Screen layouts
- Test on many devices!
Challenge 4: Sound

- No support for Sound in MIDP 1.0—must use Vendor APIs

- Handset capabilities vary from single tone, to simple polyphonic tone generator, General MIDI and MP3 or other wave format

- Possible tactics:
  - Provide sound assets in many forms
  - Use a portability layer to access Vendor APIs
Challenge 5: Portability at Deployment

- Fully portable games will be too large!
- Need a range of builds to cover available handsets
- Each must be tested on each handset
- Possible tactics:
  - Build time configurability
  - nGame Wireless Toolkit to assist in portability
Challenge 6: Commercial Return

- No copy protection with the J2ME platform
- No license management on handset
- Additional support needed for revenue-generating mechanisms—e.g., posting of high-scores, downloading of the next level
Thomas Landspurg
Chief Technical Officer
In-Fusio
About In-Fusio

- Provide Game Services for operators
- Wap, SMS, J2ME platform-based and ExEn games
- Deployed with 14 operators in Europe and Asia
- Active member of the JSR-118 (MIDP 2.0) expert group
- Contact: thomas.landspurg@in-fusio.com
Wireless Game Challenges

- Platform differences:
  - Screen size/Colors
  - CPU
  - Memory

- Implementation differences
  - As game requirements are extremely high, some small differences in VM or API implementations can generate a big difference in resulting game

- Business models differences
  - Same game may use different BM depending on the operator
Huge issue:

- How can you ensure that a Game run with the same quality on all devices?
- Assuming hundreds of Java™ technology-based devices, can you physically test the game on all those devices (and in all the languages)?
  - Use risk management instead of full testing
  - Use common building blocks, and test these blocks on all target devices
Some Solutions

- Take in account multiple size/platform issues from the beginning (Game Design phase)
- Create some building block library, tested on a lot of platform
- J2ME platform improvement (JSR-118 and others) provide some solutions for games
- Use server to solve some of these issues
  - Dynamic jar file creation on server side
J2ME Platform, Game API: Provide Some Solutions

- Speed! High level functions for games
- Behavior of such API will be tested with TCK -> increased game interoperability
- Reduction of development time and game quality increase
- Reduce downloaded code size
Make the difference between a Wireless game and an embedded game (or GameBoy game)

With an end-to-end, Java technology-based solution, it’s easiest to do:

- Server side dynamic creation of downloaded content
- Multiplayer games (same libraries or convention can be used on both sides)
- Extensions: server-side “game logic”
Future Directions

- Networking
  - Provide game-oriented multiplayer facilities, independent of the underlying protocol
  - Lobby
  - Presence

- Game-oriented graphic features
  - More 3D
  - Vector animation
  - Real-time graphic transformations
New J2ME Wireless Toolkit Features

- Network monitor
- Memory monitor
- Method call profiler
- Performance simulation
- MIDlet suite size reduction (through obfuscation)
Network Monitor
Memory Monitor
Memory Monitor (Cont.)

The Memory Monitor Extension shows a list of objects along with their statistics. The table includes columns for Name, Live, Total, Total Live, and Average. The objects are listed with their associated values and percentages. The current session shows 597 objects used 25460 bytes, free 474532 bytes, and total 500000 bytes.
# Method Call Profiler

![Method Call Profiler](image)

## Call Graph

- **(100.0%)** <root>
- **(6.68%)** example.manyballs
- **(0.11%)** java.lang.Thread
- **(4.13%)** javax.microedition.io.Displayable
- **(73.81%)** com.sun.awt.Midp
- **(14.43%)** com.sun.midp.Main
- **(0.05%)** java.lang.Class

## All calls under example.manyballs.SmallBall.run

<table>
<thead>
<tr>
<th>Name</th>
<th>Count</th>
<th>Cycles</th>
<th>%Cycles</th>
<th>Count With Childs</th>
<th>%Count With Childs</th>
</tr>
</thead>
<tbody>
<tr>
<td>example.manyballs.SmallBall.run</td>
<td>0</td>
<td>104549</td>
<td>1.4</td>
<td>413573</td>
<td>5.6</td>
</tr>
<tr>
<td>javax.microedition.io.Canvas.repaint</td>
<td>984</td>
<td>25115</td>
<td>0.3</td>
<td>300828</td>
<td>4.1</td>
</tr>
<tr>
<td>javax.microedition.io.Displayable.repaint</td>
<td>984</td>
<td>19941</td>
<td>0.2</td>
<td>276513</td>
<td>3.7</td>
</tr>
<tr>
<td>javax.microedition.io.DisplayImpl</td>
<td>983</td>
<td>28659</td>
<td>0.3</td>
<td>255572</td>
<td>3.5</td>
</tr>
<tr>
<td>com.sun.awt.midp.EventHandler.schedule</td>
<td>85</td>
<td>226913</td>
<td>3.1</td>
<td>226913</td>
<td>3.1</td>
</tr>
<tr>
<td>java.lang.Thread.sleep</td>
<td>984</td>
<td>8396</td>
<td>0.1</td>
<td>8396</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Summary

- Choose a design strategy to address large variety of devices
- Test on physical devices as well as emulators
- Keep in mind your target business model when developing a game application
- MIDP 2.0 (JSR-118) will provide an improved gaming environment
The Java™2 Platform, Micro Edition (J2ME™) is a viable platform to develop and deploy wireless games. Developing games with J2ME technology requires a good design methodology, focus, and creativity.