

Java for Mobile Devices

Maik Hassel
hassel@simplyefficient.ca
(Simply Efficient, 2006)

J2ME in 20min?

Simply Efficient | A Bit of History

- Early 1990's: "Green Team" wanted to build robust, reliable applications on consumer devices
- Hardware wasn't advanced enough
- Java got repurposed

Simply Efficient | Things Changed...

- Consumer devices (cell phones/PDA) feature
- High resolution screens
- Powerful processors
- Lots of RAM
- Wi-Fi networking
- Sound
- Etc...

Simply Efficient | Back To The Roots

- Java 2 Micro Edition (J2ME)
- Resource-constrained JVM and a set of Java APIs
- J2ME applications: *MIDlets*

Simply Efficient | J2ME Structure

- **Configuration** defines the base platform for a set of devices with similar capabilities (memory, CPU speed, etc).
 - "Generic" (not targeted at any particular type of device)
 - Defines the core language and Java VM features that can be supported on a range of devices, along with a very small set of essential class libraries.
- **Profile** provides additional libraries
 - geared toward a particular kind of device
 - Example: MIDP provides user interface (UI) toolkit optimized for devices with small, bitmapped displays.

Simply EFFICIENT Configuration

- **"High-end"** consumer devices
 - CDC (Connected Device Configuration)
 - Typical devices have at least 2-4 MB of RAM
 - Persistent, high-speed network connectivity
 - This class includes set-top boxes and higher-end PDAs
 - More feature rich JVM
- **"Low-end"** consumer devices
 - CLDC (Connected, Limited Device Configuration)
 - Typical devices can have as little as 128K of RAM
 - Low-speed or intermittent network connections
 - Includes typical cell phones, two-way pagers and lower-end PDAs
 - JVM that it provides is very limited and supports only a small number of traditional Java classes
 - JVM is called KVM

Simply EFFICIENT Profiles

- CLDC: **MID** profile minimizes both memory and power required
 - Provides the basic API for creating applications
 - Example: javax.microedition.lcdui package
 - Allows to create the GUI elements that can be shown on a (limited) device running the MID profile on top of a CLDC configuration.
- CDC devices get their own set of profiles
 - Foundation profile
 - Personal profile
 - etc

Simply EFFICIENT Optional Packages (on top of CLDC)

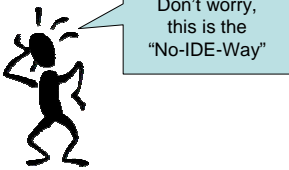
- Information Module Profile (JSR-195)
- Java Technology for Wireless Industry (JSR-185)
- Mobile Media API (JSR-135)
- Wireless Messaging API (JSR-120)
- Location API for J2ME (JSR-179)
- SIP API for J2ME (JSR-180)
- Security and Trust Services API for J2ME (JSR-177)
- J2ME Web Services (JSR-172)
- Mobile 3D Graphics (JSR-184)
- Bluetooth API (JSR-82)

Simply EFFICIENT How To Install The JVM?

- We don't!
- Device manufacturers package their devices with JVM (and associated APIs)
- Developers only need to develop applications and install them.
- Is it really that easy?

Simply EFFICIENT Lets Do It...

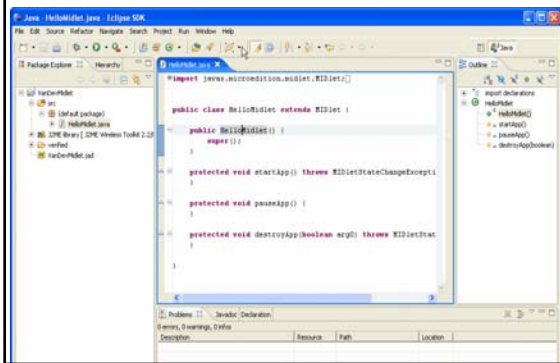
- J2ME applications are *"lean and mean"*
- Get the wireless toolkit
- Get EclipseME (J2ME plugin)
- Seven steps in the creation of a *"MIDlet"*:
 1. Designing
 2. Coding
 3. Compiling
 4. Pre-Verification
 5. Packaging
 6. Testing
 7. Deployment.



Simply EFFICIENT Step 1: Design

- Handsets impose limit on .jar file size
 - Nokia series 40 (1st Ed) has a .jar file size limit of 64kB.
- Heap sizes can be very small, around 200kb.
 - Use as few objects as possible
 - Delete references when they are no longer needed.
- Programs can not expect more than 20kb persistent storage.
 - Don't keep more than you need.
 - Newer phones support additional APIs for file storing
- Very different screen dimensions, orientations and color depths.
- Not all devices have keypads or pointers.
- Key layouts may vary too.
- Etc etc

Step 2: Code



Step 3: Compile

Same same, but different: Change the **boot CLASSPATH** while compiling MIDlets

```
javac -bootclasspath  
.. \lib\cldcapi11.jar;.. \lib\midpapi20.jar
```

Step 4: Preverify

- JVM verifies byte code before it runs any class
 - If verification fails, class file is rejected, JVM shuts down
- Too resource intensive for J2ME devices
- Thus: pre-verification process
 - Add special information to classes
- Makes the process on the device much more efficient
- The Wireless Toolkit comes with a preverification tool

Step 5: Package

- Create a special Manifest file
- Create the JAR file including the preverified *classes and* the Manifest file.
- Create the Java Application Descriptor (JAD) file.



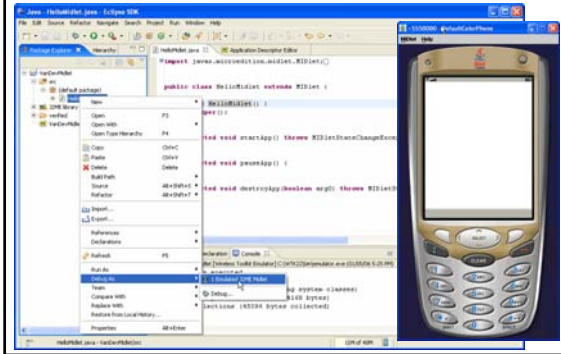
Step 6: Test

- Using a *base common emulator device*
- Mimics the functionality of an actual device on your computer.
- Part of the Wireless Toolkit
- Provides functionality that is sure to be present in the majority of devices for which the MIDlet is targeted.

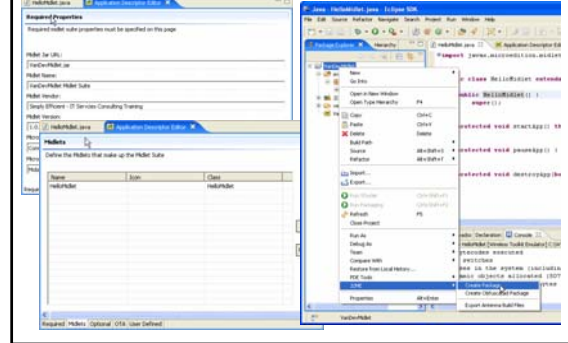
Step 7: Deploy

- Direct connection to device
 - USB, Bluetooth, ...
 - Supported by most java enabled devices
- Via web browser

Testing in Eclipse



Packaging in Eclipse



Java-Based Mobile Games

- Exp. # of mobile gamers: 220 million (2009)
- Three categories of mobile games
 - Embedded games: Soon be obsolete
 - SMS games: Expensive
 - Browser based: Most popular
 - solo or multiplayer games
 - network games
 - offline games
 - arcade games
 - etc

Java-Based Mobile Games

- javax.microedition.lcdui
 - Control elements
 - Similar to Swing
- javax.microedition.lcdui.game
 - Classes for creating/controlling game elements
 - game canvas
 - Sprites
 - Layers
 - Sound
 - etc

Java-Based Mobile Games

- Support for threads
- Support for generic keys (up, down, fire, etc)
- 2D and 3D support



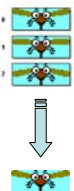
Example: CommandListener

```
public class VanDevGame extends MIDlet
    implements CommandListener
```

- Reacts to user interactions
 - soft buttons
 - Device menu item
 - Speech input, etc
- EXIT, CANCEL, BACK, OK, STOP...


Simply EFFICIENT **Example: Sprites**

- Basic visual element of a game
- Multiple frames can animate the Sprite
- Built in functionality for
 - Hide/show
 - Move
 - Transforming
 - Rotating the sprite
 - Collision detection



Simply EFFICIENT **Mobile Java3D Resources**

- 3D Graphics for Java Mobile Devices
<http://www-128.ibm.com/developerworks/wireless/library/wi-mobile1/?ca=dgr-lnxw01Java3DMobile>
- Sony Ericsson Developer World
http://developer.sonyericsson.com/site/global/newsandevents/campaigns/java_3d/p_java3d.jsp
- Sony Ericsson Mobile Java 3D forum
<http://developer.sonyericsson.com/>



I hear...I forget
 I see...and I remember
 I do...and I understand

