DSDP Mobile Tools for Java Webinar

Craig Setera
Christian Kurzke
Diego Sandin
David Marques
Gustavo de Paula
Agenda

- DSDP Mobile Tools for Java Overview
- Project History
- MTJ Concepts
- Tool vs. Framework
- MTJ as a Tool
  - Import JavaME SDK
  - Create Project and MIDlet Templates (New)
  - Application Descriptor Editor
  - Run / Debug MIDlet
  - Multi-configuration
  - MIDlet Localization
  - JMUnit Test
  - MIDlet Signing (New)
- MTJ as a Framework
- Conclusions
DSDP MTJ Overview

- Glue between the Eclipse IDE and the JavaME SDKs (emulators)
- MTJ does NOT provide the SDKs
- Each vendor provides their own SDKs
- MTJ provide the basic developer workflow
  - Import JavaME SDK (each SDK has several devices associated to it)
  - Create MIDlet Suite Project
  - Build Project
  - Create Deployment Package
MTJ Timeline

- **EclipseME 1.7.8** • Import from Netbeans

- **MTJ 0.9** • EclipseME Based • Re-organize code with Eclipse code standards • Setup build environment • Import from Netbeans • Import from EclipseME • Enhanced JAD Editor • Bug fixes • Library Support • MIDlet Localization • Multiple Device Support

- **MTJ 0.9.1** • MIDletTemplates • SDK • Join Galileo Train • Re-structure code (increase flexibility) • Adapt code from MTJ 0.7 • Signing • Deployment • Packing

- **MTJ 1.0**

- **MTJ 0.7**

**Quarterly Milestones:**
- **Q3/2008**
- **Q4/2008**
- **Q2/2009**
MTJ Concepts
The DeviceImporter knows how to “talk” to one SDK

- Returns all Devices that one SDK have
- Each device has a DeviceClasspath associated to it
  - ... and a list of APIs
- Devices are persisted on the DeviceRegistry
MTJ Concepts: Symbols

- A **Symbol** is a pair <name; value>
- Symbols are organized in **SymbolSets**
- Each SymbolSet has a name and a list of Symbols
- i.e.:
  - Symbols: `<screen.width;240>, <screen.height;320>`
  - Symbol set: `<MOTV3; <screen.width;240>, <screen.height;320>>`
- **SymbolSetFactory** can import devices database and same them on the workspace
MTJ Concepts: Runtimes

- A MTJ project has a list of MTJRuntimes
  - Each project has a current **active Runtime**
- Each Runtime has
  - A Device
  - A project symbol set (created based on the device properties and APIs)
  - A list of workspace SymbolSets
MTJ Concepts: MIDlet Projects

- MTJ targets JavaME development
  - Different runtime platforms (MIDlet, Xlet, java class, etc.)
  - Currently only MIDlets are supported on MTJ

- MTJ defines a new project type
  - MIDlet Suite Project

- This project enables
  - Create MIDP deployable JAR/JAD files
  - JAD Editor
MTJ Concepts: JavaME Perspective

- Once a MIDlet Suite project is created, eclipse switches to JavaME perspective
- Similar to a usual Java Perspective, but with MTJ actions
MTJ Features
Framework vs. Tool

- Who is the **main** target user of MTJ?
  - Tool Developer: needs a mobile development framework
  - JavaME Developer: need a mobile development tool

- Original MTJ Approach seems to be **good** for *Tools developer*, but not for JavaME developers

- EclipseME is **pragmatic**, targets JavaME developers at the expense of some flexibility

- Final MTJ must be
  - “**Framework enough**” that a company can extend to create its own tool
  - “**Tool enough**” that can be used by a mobile application developer as it is
MTJ as a Tool

- **Target User**: MIDlet Suite Developer

- Provide **all** necessary support to develop **MIDP-based application**
  - Import SDK
  - Create Project / MIDlet Wizards
  - Edit MIDlet Suite Application Descriptor
  - Build MIDlet Suite Package
  - Run / Debug MIDlet
Tool: Import SDK

- Any UEI or Java SE-based SDK
- User can select which device to be imported

- Devices are added to Device Management list
- User is able to select one as the default
Tool: Creation Wizards

- Select JAD Name
- Select Current SDK & Device
- Enable Preprocessing on that Project
Tool: MIDlet Templates (under development)

- Different Template are available on MTJ
  - Templates can be selected on the MIDlet creation wizard
  - There is also an extension point to add new templates
Tool: Application Descriptor Editor
Tool: Run / Debug MIDlet

Overview

Required Information
This section describes required information about this application.

- MIDlet Jar URL: MyFirstMIDlet.jar
- MIDlet Name: MyFirstMIDlet MIDlet Suite
- MIDlet Vendor: MIDlet Suite Vendor
- MIDlet Version: 1.0.0
- Microedition Configuration: Connected Limited Device Configuration (1.1)
- Microedition Profile: Mobile Information Device Profile (2.0)

Running
Run your application within a Java ME device:
- Launch as emulated Java ME MIDlet
- Launch as emulated Java ME JAD

Debugging
Debug your application within a Java ME device:
- Launch as emulated Java ME MIDlet in Debug mode
- Launch as emulated Java ME JAD in Debug mode

Runtime
Specify the execution environments to run this Project:
- SDK: MOTODEV SDK for Java(TM) ME
- Device: GENERIC

Packaging
Package your application:
- Create package
- Create obfuscated package

Exporting
Generate your build files based on the configuration of the MIDlet Project:
- Export Antenna Buildfiles

DSDP Mobile Tools for Java Webinar | © 2009 by Motorola and others; made available under the EPL v1.0
Tool: Multi Configuration

- Device fragmentation solution
  - Preprocessing support
  - Multiple devices supported on each project
  - Preprocessing Code assist
Tool: MIDlet Localization

- Enabled MIDlet localization based on current device language
- Generates **localized property files** and code to access them on the MIDlet
Tool: JavaME Unit Test

- Create and Run Test Cases on an Emulator
- Create final deployable MIDlet with Test Framework to run on target Devices
Tool: Signing Enhancements (under development)

- Keystore management
- Automatically scan code to identify protected APIs
- Sign MIDlet Suite
MTJ as a Framework

- Original objective of MTJ 0.9 is to provide a tool for JavaME developers
  - Framework is not a primary focus, but a welcome “side effect” of proper design

But...

- MTJ API is one of the main objectives on 1.0 release
  - Code cleaning
  - Add some new extension points
  - Publish only the code that make sense
Framework: MTJ API 1.0 (under development)

- Not initial MTJ Focus. But a desirable result

- Current API is based on original EclipseME API
  - A lot of cleaning
  - Move classes / interfaces to internal packages
  - Leave as public only the minimum set of APIs that we are sure are necessary
  - Add some new APIs requested by the community
Framework: MTJ API 1.0 (under development)

- 9 extension points
- Focus to provide SDK Providers extensions
- Final list of APIs are still under discussion
- Details specification of each API is still under discussions
Conclusions
Conclusions

- MTJ 0.9.1 already have several new features available to MIDlet developers

- A couple of more features are under development and will be available on MTJ 1.0

- MTJ API will enable the definition of a framework that can be extended by different vendors

**Help is always welcome!**
Thanks!

DSDP MTJ Web site: www.eclipse.org/dsdp/mtj
DSDP MTJ Wiki: http://wiki.eclipse.org/DSDP/MTJ
Newsgroup: news://news.eclipse.org/eclipse.dsdp.mtj
Developer List: http://dev.eclipse.org/mailman/listinfo/dsdp-mtj-dev