Building a Reliable Remote Communication Device with Java ME 8

JavaOne San Francisco 2014 - CON2285

Leonardo Lima – V2COM (llima@v2com.mobi, @leomrlima)

About Leonardo Lima

- Software Development Manager at V2COM
- V2COM representative at the JCP-EC: Java Community Process – Executive Committee
- Spec Lead - JSR 363 – Units of Measurement

- llima@v2com.mobi and/or @leomrlima

V2COM

- V2COM is a leading Latin American provider of Smart Grid technologies and Advanced Metering Infrastructure (AMI). Our offer includes hardware, software and services that can reduce losses and increase water and energy efficiency, currently connecting more than 1.000.000 devices.

Agenda

- The definition of **reliable**
  - What it means to be a reliable device?
- Java ME 8 improvements
  - What Java ME 8 brought to the game
- Java ME 8 applied
  - Improvements, applied
- More improvements to come
  - New JSRs and room for improvements

The definition of **reliable**

What it means to be a reliable device?

What does **reliable** mean?

- **re·li·a·ble** (adjective): *that may be relied on; dependable in achievement, accuracy, honesty, etc.*

- Do not fail!
- Do not lose data!
- Do not let others tamper with your data!

Do not fail!

- Protected against the environment
- Protected against hardware failures
- Protected against software failures
- Protected against attackers

Do not lose data!

- It’s the device mission to ensure the data acquired is safely stored in itself.

- It’s the device mission to ensure that data is safely transferred to the backend services.

Do not let others mess with your data!

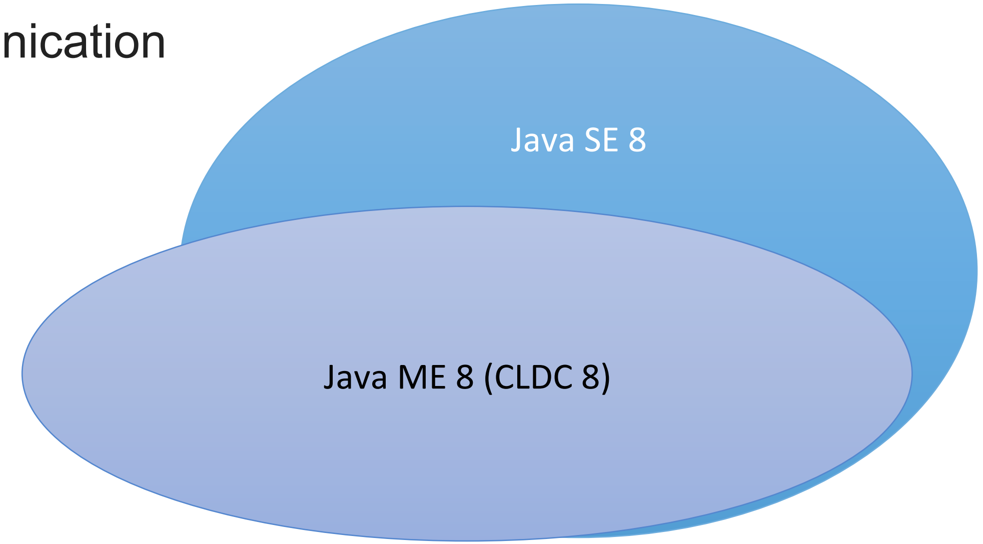
- Take into account all security aspects
- Authenticity: data you sent is sent from you, really
- Confidentiality: Only you and the *right* server sees the data
- Integrity: data received == data sent

Java ME 8 improvements

What Java ME 8 brought to the game

Improvements

- Language Improvements (less bugs!)
- Service Loader pattern
  - Events and Inter-Midlet Communication
- Device I/O
- Security Improvements
- Shared Libraries



Shared Libraries (LIBlets)

- Can be uniquely updated, disabled/enabled.
- A shareable software component that one or more applications MAY use at runtime
- Save static footprint size by enabling multiple application suites to share the same common code without packaging them redundantly
- Reduced download times for applications that declare dependencies on shared components

Service Loader

- Enables creation of microservices inside your device
- Independent services that can be uniquely updated, disabled/ enabled.
- Monolithic firmware is now split into different services

Events

- Object driven communication
- Publisher/Subscriber pattern
  - One-To-Many communication
  - Asynchronous delivery
- Caveat: Events are (silently) discarded if there are no subscribers

Inter-MIDlet Communication

- Raw, bidirectional socket communication
  - Fast and Synchronous!
- Reliable one-to-one communication
  - You know when the peer is there
- Just like “localhost” sockets

Network Security

- TLS protocol enhancements
  - SecureConnection and SecureServerConnection support TLS 1.2
  - SecureServerConnection provides the server-side of a TLS connection
- DTLS protocol support for TLS over UDP
  - SecureDatagramConnection provides client-side support for DTLS

Application Provisioning & Management

- **Application Management System (AMS)** allows fine grained control on what get installed
- **App Management Agent** manages local applications via the AMS api

V2COM

Device I/O

- Unified, object-oriented way to interact with I/O
- Basic I/Os already mapped
  - I2C, GPIO, SPI, UART, ADC, Watchdog, PWM, Pulse Counter
- Architecture is ready for extension with generic AbstractDevice, DeviceProvider

Language improvements

- Improved collections
- Generics
- Autoboxing
- Static imports
- Enumerations
- Try with resources
- Better byte and string manipulation
- Events
- Logging

Java ME 8 applied

Improvements, applied to our use cases

Before…

- We had 3 big blocks of code, merged as one on deploy
  - Hardcore for hardware abstraction
  - Zion for commons infrastructure and AT
  - App for application logic

- One big file to manage (OTAP)
- Java 1.3 language…
  - Did I mean Latin?

… and now

- Multiple services and libraries
  - Instead of big libraries
- Less boilerplate code
  - Enum, foreach, try with resources makes it less verbose, more agile
- Less ancient code
  - Easier for “desktop developers” to understand and join the fun
- Less “made in home” code
  - Many pieces are now already in the platform, like Device I/O, Events, Logging
  - Less code for us to mantain

Shared Libraries

Crypto / Hashes

Queues

Advanced String & Byte Manipulation

Units and Measurements

Advanced Math

Concurrency

Device protocols

Event and Bus

Services

(Safe) Storage	Black Box	Logging	Watchdogs
Data acquisition (Application)	Remote transfer	System Configuration	Power management
External Sensor	WAN Management	Resource Management	Firmware Update Management

More improvements to come

New JSRs and room for improvements

JSR 363 – Units of Measurement

- JSR to beter type data that we acquire, targeting ME 8
- https://java.net/projects/unitsofmeasurement

```
import javax.measure.quantity.*;
public class TemperatureRecord {
    private Time timestamp;
    private Temperature temperature;
    private String sensor;
...getters, setters...
}
```

JSR 363 – Units of Measurement in JDK

- Otávio Santana (SouJava, OpenJDK) is actively working in a SE 8 port of the JSR!
- Streams, Lambda, High precision (BigDecimal): guaranteed!
- Reducers, sorters, summation, group and filters.

```
List<Quantity<Time>> sortNaturalList = times
     .stream()
     .sorted(QuantityFunctions.sortNaturalDesc())
     .collect(Collectors.toList());
```

Room for improvement

- Configuration
- Concurrency utilities
- Standard protocols (MQTT, CoAP, JSON, HTTP, REST)

Q?

Thanks!

llima@v2com.mobi

@leomrlima

www.v2com.mobi