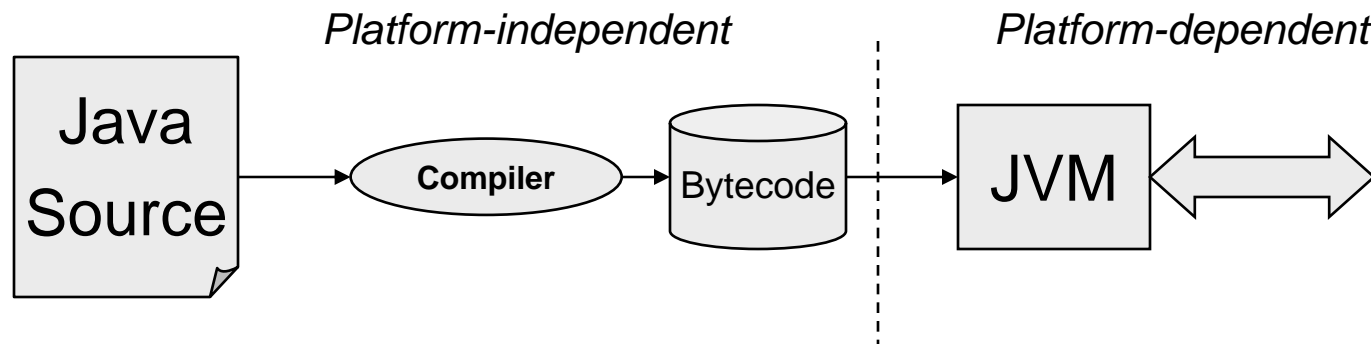

EECE 276

Embedded Systems

Survey: Java/RTJ, Occam2
Coding standards

Java Language

- Modern, OO, platform-independent language
 - » Platform independence through JVM



- Direct support for threads and (simple) synchronization
 - » Focus on GUI-s and web apps

Java Language

- Simpler and easier to read than C++
 - » No preprocessing
 - » Class / module / file structure (~package)
- No pointers
 - » All objects and arrays passed by reference
 - » Boundary checks on arrays
 - » Automatic memory management on objects/arrays
- OO: classes (only) + No standalone functions
- Interfaces: class interface specs
 - » Only interface, no implementation

Java Language

- No multiple inheritance
 - » A class can implement multiple interfaces
- Strings as built-in objects
- No automatic (type) coercion
 - » Implicit “upcasting”, “downcasting” requires code
- Threads as built-in elements
 - » synchronized classes
 - » Signals (wait/signal)
 - » Support for distributed programming

Real-time Java

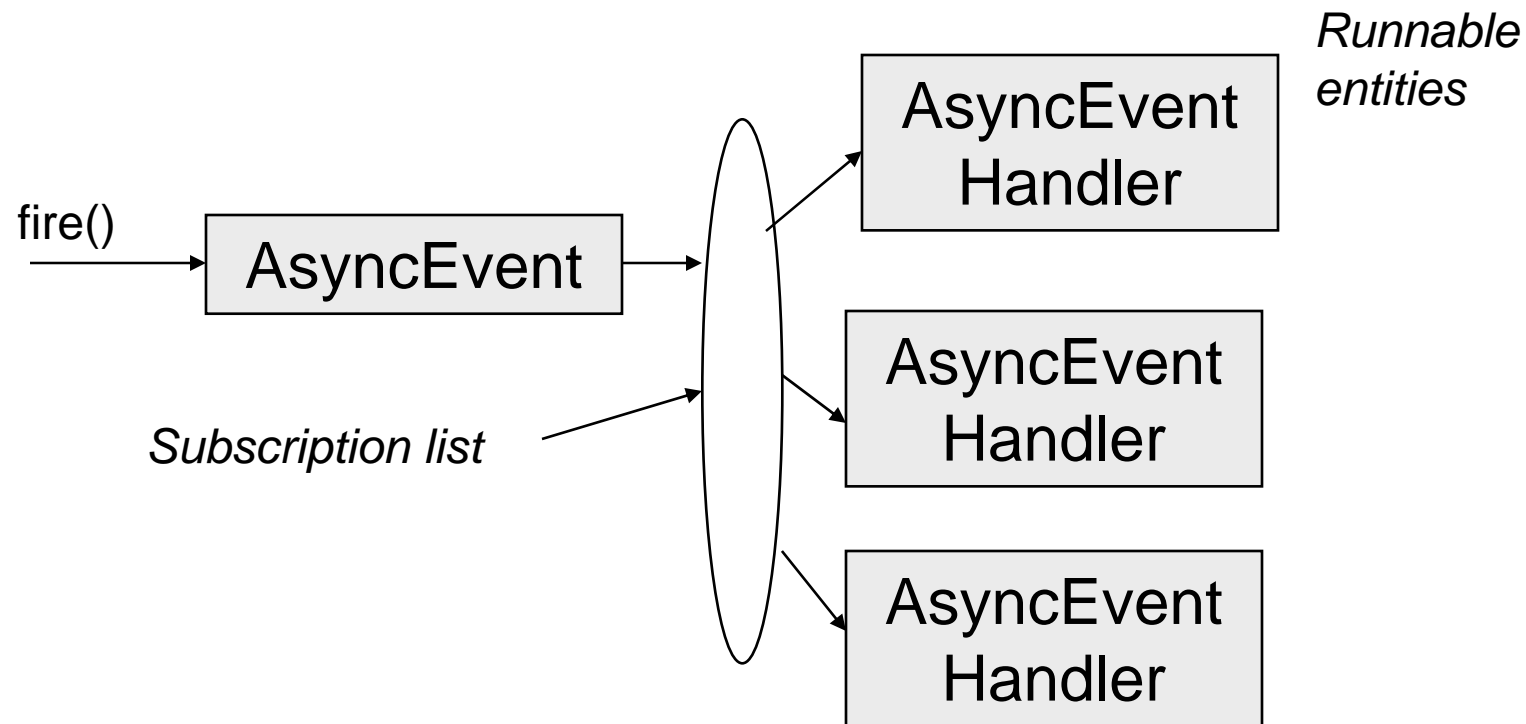
- Extensions to Java to address specific embedded system needs
 - » www.rtj.org
- Memory management
 - » Garbage-collected heap (like Java)
 - » Short- and long-lived objects outside of heap
 - » Memory pools
 - » Alter GC algorithm
 - » Access to raw memory (bytes/words @ address)
 - » Access to physical memory (objects @ address)

Real-time Java

- Scheduling
 - » Real-time threads (with or w/o heap)
 - » Priority ordered queues for all threads waiting for resources (CPU, locks, signals)
 - » Priority inheritance protocol by default
 - » Fine-grain control over scheduler behavior
- Time and timers
 - » Calculations with Time
 - » Clock/Timer/PeriodicTimer/OneShotTimer

Real-time Java

- Event-driven processing:
 - » AsyncEvent and AsyncEventHandler



Real-time Java

- Summary
 - » Features for real-time programming
 - » Low-level abstractions for controlling real-time behavior
 - » Similar to services provided by real-time kernels
 - » Some literature, some applications

Occam2

- Simple, high-level language with direct support for distributed systems
 - » Direct support for processes
 - » Process primitives:
 - send and receive (through port)
 - **seq**, **par**, and **alt** control structures
- Supported by the Transputer
 - » High-performance RISC processor with 4 high-speed serial ports
 - » HW Implementation of CSP (Hoare, 1978)
- Historical significance, although some concepts live on
 - » High-speed communication ports integrated with the processor

Coding standards

- Rules for writing code (such that others can read it)
- Rules for
 - » File headers
 - » Naming conventions
 - » Commenting conventions
 - » Formatting conventions
 - » Programming constructs (to be used/not used)