Java
Virtual Developer Day—Java 2014
Get the latest information about Java from the experts: Java SE 8, Java EE 7, and Java Embedded. Learn about Lambdas and more!
Brought to You by Oracle Technology Network
ORACLE

Virtual Developer Day: Java 2014 – May 28th, 2014

10:30 AM - 2:30 PM IST / 1:00 PM - 5:30 PM SGT / 3:00 PM - 7:30 PM AEDT

Agenda

Time (IST)			
10:15 a.m.	Event Platform Opening		
10:30 a.m.	Keynote - Java: Present and Future		
	Java EE 7	**Java SE 8**	**Java Embedded**
11:00 a.m.	Java API for WebSocket + Demo (11.00-11.40)	55 New Features in Java SE 8 (11.00-12.00)	Coffee with Dessert: Java SE & ME 8 Embedded and the Raspberry Pi (11.00-12.00)
11:40 a.m.	Java API for JSON Processing + Demo (11.40-12.20)		
12:20 p.m.	JMS 2.0 (Java Message Service)	Things you should know when developing JavaFX applications (12.00-1.00)	Using Oracle Event Processing for Oracle Embedded Java to Make Big Data Smaller (12.00-1.00)
12:40 p.m.	JAX-RS 2.0 (Java API for RESTful Web Services)		
1:00 p.m.	Break and Dedicated Java Community Lounge Time		
1:15 p.m.	Batch API for the Java Platform + Demo (1.15-2.00)	Lambda Expressions Tutorial (1.15-2.00)	Talking to the Real World With the Device Access API (1.15-2.00)
2:00 p.m.	Concurrency Utilities for Java EE + Demo (2.00-2.30)	Up and Running with NetBeans IDE (2.30-2.30)	Big Tools for Small Devices (2.30-2.30)
2:30 p.m.	Event Close		

Abstracts

Keynote – Java: Present and Future

With the release of Java EE 7, Java SE 8 and Java ME 8 there have been significant enhancements in functionality across the Java platform. Java will, however, continue to evolve to meet the needs of developers across the globe and much of this will be through community involvement. This session will review the status of the Java platform as it stands today and look at some ideas for the future of the platform. We'll also discuss the many different ways developers can be part of the community shaping the future of Java.

Java EE 7 Track –

Java API for WebSocket + Demo

The Java API for WebSocket (JSR 356) is a wholly new technology for the Java EE platform. This rapidly adopted network protocol that is a foundational piece of the HTML5 standard gives Java EE developers the ability to add efficient, asynchronous bidirectional messaging into their web applications, enabling them to make their web applications more active and engaging. This session will work through the highlights of this new Java API in Java EE 7 so that you will see exactly what features it has and how to integrate it into your Java EE application. The second half of the session will illustrate the WebSocket API with different demos (annotated and programmatic based).

Java API for JSON Processing + Demo

JSON is a lightweight data exchange format used increasingly in RESTful Web services by highly visible Websites (Facebook, Twitter, Amazon, and the like). This session will show how JSR 353: Java API for JSON Processing can be used to process JSON. It goes over streaming API to produce/consume JSON, and also object model API to build a Java object model for JSON. This session will be concluded by demos to illustrate use of JSON-P API in JAX-RS and other applications (for e.g. to process Twitter search results).

Java Message Service (JMS) 2.0

This session introduces the new features of JMS 2.0 (JSR 343) and explains how they improve ease of use and make developers more productive. JMS 2.0 introduces a completely new "simplified" API which means fewer objects to manage, simpler methods to call and less code to write. In Java EE, the use of annotations and dependency injection reduces the amount of code needed still further. Improvements have also been made to the existing "classic" API to make it simpler and easier to use. The session also includes a summary of some of the new messaging features which have been added to tackle real-world issues such as scalability and error handling.

Java API for RESTful Web Services 2.0

JAX-RS is the Java API for building RESTful web services. This session focuses on the new features introduced in JAX-RS 2.0 (JSR 339) including the client API, filters and entity interceptors, asynchronous processing and hypermedia. These new features are centered around providing ease of use and reduced development time for your applications. Additionally, application portability is greatly increased by minimizing the need to use proprietary extensions in existing JAX-RS implementations. This session briefly introduces the new features and shows code snippets of the API for each one.

Batch API for the Java Platform + Demo

Batch applications: anachronism or enterprise workload? Asynchronous bulk (or long-running) processing, historically called "batch", has been quietly co-existing along-side online applications for years. In recent years, more and more of this batch processing has been written in Java. There has never been an API standard for these applications - until now! Batch Applications for the Java Platform (JSR 352) ushers in the world's first Java standard for building batch applications. In this session you will gain a basic understanding of JSR 352, including its motivation, feature highlights, selected API overview, job scheduling language introduction, and key use cases. The second part of the session will demonstrate some key Batch API features such as the checkpoint contract, property injection, JSL (Job Specification Language), resuming failed job, ...

Concurrency Utilities for Java EE + Demo

Concurrency Utilities for Java EE (JSR 236) is a new addition to Java EE 7 platform. It provides a simple and standardized mechanism for Java EE application developers to easily and safely add asynchronous capabilities to their applications. This session provides an overview of what is provided in this JSR. A demonstration showing how to use this API and its different executors services (ManagedExecutorService, ManagedScheduledExecutorService, and ManagedThreadFactory) will conclude the session.

Java SE Track

55 New Features in Java SE 8

Java SE 8 is the next release of the core Java platform and contains lots of exciting new features. In addition to the big features like Lambda expressions, extension methods for interfaces and a new Date and Time API there are plenty of smaller features as well. This session will rapidly cover fifty-five new features that are scheduled for inclusion in Java SE 8 when it is released early in 2014.

Things you should know when developing JavaFX applications

Developing a JavaFX application in principle is not very different from developing a Swing application. The things you have to know are related to the new possibilities of JavaFX like Properties, Bindings and CSS support. Because styling a JavaFX application is completely different from styling a Swing application this session will show some tips on how to give your JavaFX application a custom look. In addition it will be shown how to make use of FXML and SceneBuilder when creating JavaFX applications. Other interesting things like using additional scenes and stages within your app and creating custom controls by extending existing controls will be covered. The session will try to show different concepts that will help you when developing applications in JavaFX.

Understanding Lambda Expressions in Java

The big language features for Java SE 8 are lambda expressions (closures) and default methods (formerly called defender methods or virtual extension methods). Adding lambda expressions to the language opens up a host of new expressive opportunities for applications and libraries. You might assume that lambda expressions are simply a more syntactically compact form of inner classes, but, in fact, the implementation of lambda expressions is substantially different and builds on the invokedynamic feature added in Java SE 7.

This session will explain the ideas behind lambda expressions, how they will be used in Java SE 8 and look at some of the details of their implementation

Java and JavaScript: Project Nashorn

The JVM knows how to execute bytecodes, but it doesn't care where they come from. Compiling JavaScript to bytecodes has specific challenges in terms of how to optimise performance, as the JVM was designed for a statically typed language. The Oracle Nashorn JavaScript project was created to provide high performance execution of JavaScript by the JVM. This presentation goes through the performance work that has gone on to make JavaScript-to-bytecode generation for execution on the JVM feasible. It shows that the new invokedynamic bytecode gets us part of the way there but may not quite be enough. What other tricks did the Nashorn project use? The presentation also discusses future directions for increased performance for dynamic languages on the JVM, covering proposed enhancements to both the JVM itself and to the bytecode compiler.

Streams and Functions for Simpler Concurrency in Java SE 8

Lambda expressions in Java provide a simple and powerful new way to avoid the use of anonymous inner classes with single abstract method types. On their own, however, they do not provide the developer with a significant reduction in complexity for many tasks. Some of the most powerful features being introduced in Java SE 8 are enhancements to the class libraries in the form of java.util.stream and java.util.function. These essentially introduce functional-like programming constructs to Java.

This session will look at the new libraries and how they can be used with Lambda expressions to simplify common coding tasks. We'll also look at how the use of these libraries means developers no longer need to worry about writing code that is inherently serial or parallel. A simple method call change is all it takes to switch between the two.

Up and Running with NetBeans IDE

Without tools, the new features provided by Java SE 8 and JavaFX 8 would be really hard to use! For example, wouldn't you like to upgrade all your Java applications to use Java SE 8 language features, all in one go? A range of code analyzers and supporting features, such as code completion and code hints, as well as templates, wizards, and samples are needed so that you can get started quickly and also be productive once you are familiar with lambda expressions, streams, functions, and the many great new features in JavaFX, such as 3D support. This session will cover a wide range of new features right across NetBeans IDE 8 to support the technologies scheduled for inclusion in Java SE 8 and JavaFX 8.

Java Embedded Track

An Introduction to Embedded Java

We've been talking about the 'Internet of Things' for over ten years. Now that Moore's law has reduced the cost of including processing and connectivity in everyday devices to a small fraction of the product's price this is now an economic and technological reality. Just having the necessary hardware is only half of the equation; application code is the other half. Developing embedded software is notoriously difficult, requiring high levels of expertise in low-level languages. Embedded Java makes application development easier, quicker and cheaper. In addition, the reality of 'write once, run anywhere' and the Java virtual machine makes deployment of code to multiple heterogeneous platforms simple and straightforward.

This session will look at how embedded Java (in the form of both Java SE Embedded and Java ME Embeedded) can be used to program devices using everything from microcontrollers with a few hundred kilobytes of memory to those with as much computing power as a desktop. Building on this we will look at how Java ME 8 Embedded, Java SE 8, JavaFX and Java EE can be combined to enable a single language to be used when developing applications from ‘device to datacentre’.

Coffee With Dessert: Java SE & ME Embedded and the Raspberry Pi

The Raspberry Pi has caused a huge wave of interest amongst developers, providing an ARM powered single board computer running a full Linux distro off an SD card and all for only $35!

After an introduction to the Raspberry Pi and the ARM architecture this session will look at how Java can be used on a device like this. Oracle have released an early access preview of JDK 8 including JavaFX and Java ME 8 Embedded tuned specifically for the Raspberry Pi. This includes a very useful Device Access API enabling the use of sensors and actuators easily from Java code using the Raspberry Pi’s external interface. Using these releases we will show a variety of demonstrations of what the Raspberry Pi is capable of. Prepare to be amazed at what this tiny board can do.

Unleash the Power of Java ME Embedded 8

The Java language and runtime platform are pervasive in the embedded space, and now with the forthcoming release of Oracle Java ME Embedded 8 Oracle provides a robust platform for the rapid explosion of programmable, connected devices to create what we call today: Internet of things.

Oracle Java ME Embedded is based on the Connected Limited Device Configuration (CLDC), and is a Java runtime stack optimized for small embedded systems. It provides a robust and flexible application platform with dedicated embedded functionality for always-on, headless (no graphics/UI), and connected devices. Oracle Java ME Embedded enables system designers and developers to create sophisticated small embedded solutions leveraging the benefits of the Java language, runtime, and ecosystem while meeting tight system resource targets.

This session will help you learn everything you need to know to get ready to program the next generation of the "Internet of Things".

Using Oracle Event Processing for Oracle Embedded Java to Make Big Data Smaller

Oracle Event Processing for Oracle Java Embedded is a solution for building embedded device applications to filter, correlate and process events in real-time so that downstream applications, services and event-driven architectures are driven by true, real-time intelligence.

This session will explain the architecture of the Event Processor and show how it can be used to reduce the amount of data being fed to a back-end data processing system by filtering and collating data before it is transmitted. Code for an example system will be discussed.

Talking to the Real World With the Device Access API

Embedded applications almost always involve the use of specialised hardware connected to a processing platform. This can take the form of both sensors and actuators, many of which use standard communications methods like GPIO, I2C and SPI. Java ME Embedded, which is targeted at small footprint embedded devices, includes a powerful API for communicating with these types of devices: the Device Access API.

This session will go through the various interfaces and APIs demonstrating how to integrate sensors like temperature and light as well as control LEDs and motors using simple Java code. The session will conclude with a more complex example of a Concept Board form Gemalto using the Device Access API and cellular technology to monitor and control a sample system.

Big Tools for Small Devices

The ease and productivity introduced by Java SE Embedded and Java ME Embedded is supported by a set of tools integrated into NetBeans IDE. For Java SE Embedded development, features have been added to NetBeans IDE 8 to deploy, run, debug, and profile Java SE applications on embedded devices, such as Raspberry PI, directly from the development environment, while simultaneously Java ME Embedded 8 development features are integrated into NetBeans IDE 8. Using NetBeans IDE 8, we will show development with the Raspberry Pi, as an example of an embedded device. Tiny boards and other embedded devices are easy and intuitive to develop on with the tools NetBeans IDE 8 makes available.