**JAVA 8**

**Duration: 3 Days**

**Detailed Course Outline**

Introducing Lambda Expressions

* Describing the purpose of an anonymous inner class
* Describing drawbacks to anonymous inner classes
* Describing the components of a lambda expression
* Defining a functional interface
* Creating programs that use lambda expressions

A Case for Lambda Expressions

* Discussing the reasons for adding lambda expressions to the Java language
* Reviewing the standard way of extracting data in Java
* Refactoring code to reduce redundancy
* Refactoring code to use inner classes
* Refactoring code to use lambda expressions
* Listing the benefits of lambda expressions

Filtering Collections with Lambdas

* Iterating though a collection with forEach
* Iterating through a collection using lambda syntax
* Describing the Stream interface
* Filtering a collection using lambda expressions
* Calling an existing method using a method reference
* Chaining multiple methods together
* Comparing function and imperative programming
* Defining pipelines in terms of lambdas and collections

Using Built in Lambda Types

* Listing the built in interfaces included in java.util.function
* Determining true or false with a Predicate
* Processing an object and return nothing with Consumer
* Processing one object and return another with Function
* Generating a new object with Supplier
* Using primitive versions of the base interfaces
* Using binary versions of the base interfaces

Collection Operations with Lambda

* Extracting data from an object using map
* Searching for data using search methods
* Describing the types of stream operations
* Describing the Optional class
* Performing calculations using methods
* Describing lazy processing
* Sorting a stream
* Saving results to a collection using the collect method

Parallel Streams

* Reviewing the key characteristics of streams
* Contrasting old style loop operations with streams
* Describing how to make a stream pipeline execute in parallel
* Listing the key assumptions needed to use a parallel pipeline
* Defining reduction
* Describing why reduction requires an associative function
* Calculating a value using reduce
* Describing the process for decomposing and then merging work

Lambda Cookbook

* Modifying a list using removeIf
* Updating a list using replaceAll
* Updating a map using computeIfAbsent, computerIfPresent, and merge
* Sending the keys and values from a map to a stream
* Reading a file to a stream
* Reading a text file into an ArrayList
* List, walk, and search a directory structure using a stream
* Flattening a stream using flatMap

Method Enhancements

* Considering the importance of building good libraries
* Using static methods in Interfaces
* Using default methods
* Understanding default method inheritance rules

Using the Date/Time API: Working with Local Dates and Times

* Listing the goals of the Date/Time API (JSR-310)
* Creating and manage date-based events
* Creating and manage time-based events
* Combining date and time into a single object

Using the Date/Time API: Working with Time Zones

* Working with dates and times across time-zones and manage changes resulting from daylight savings

Using the Date/Time API: Working with Date and Time Amounts

* Defining and create timestamps, periods and durations
* Applying formatting to local and zoned dates and times

JavaScript on Java with Nashorn: Creating and executing shell scripts

* Creating and execute shell scripts using JavaScript and Nashorn

JavaScript on Java with Nashorn: Writing JavaScript Applications

* Developing JavaScript applications that leverage Java code using Nashorn

JavaScript on Java with Nashorn: Writing JavaFX Applications Using JavaScript

* Running JavaScript script from Java applications usingJSR-223
* Prototype JavaFX applications using Nashorn and JavaScript

[Collections](http://docs.oracle.com/javase/8/docs/technotes/guides/collections/changes8.html)

* Classes in the new java.util.stream package provide a Stream API to support functional-style operations on streams of elements. The Stream API is integrated into the Collections API, which enables bulk operations on collections, such as sequential or parallel map-reduce transformations.
* Performance Improvement for HashMaps with Key Collisions

[Security](http://docs.oracle.com/javase/8/docs/technotes/guides/security/enhancements-8.html)

* Client-side TLS 1.2 enabled by default
* New variant of AccessController.doPrivileged that enables code to assert a subset of its privileges, without preventing the full traversal of the stack to check for other permissions
* Stronger algorithms for password-based encryption
* SSL/TLS Server Name Indication (SNI) Extension support in JSSE Server
* Support for AEAD algorithms: The SunJCE provider is enhanced to support AES/GCM/NoPadding cipher implementation as well as GCM algorithm parameters. And the SunJSSE provider is enhanced to support AEAD mode based cipher suites. See Oracle Providers Documentation, JEP 115.
* KeyStore enhancements, including the new Domain KeyStore typejava.security.DomainLoadStoreParameter, and the new command option -importpassword for the keytool utility
* SHA-224 Message Digests
* Enhanced Support for NSA Suite B Cryptography
* Better Support for High Entropy Random Number Generation
* New java.security.cert.PKIXRevocationChecker class for configuring revocation checking of X.509 certificates
* 64-bit PKCS11 for Windows
* New rcache Types in Kerberos 5 Replay Caching
* Support for Kerberos 5 Protocol Transition and Constrained Delegation
* Kerberos 5 weak encryption types disabled by default
* Unbound SASL for the GSS-API/Kerberos 5 mechanism
* SASL service for multiple host names
* JNI bridge to native JGSS on Mac OS X
* Support for stronger strength ephemeral DH keys in the SunJSSE provider
* Support for server-side cipher suites preference customization in JSSE

[JavaFX](http://docs.oracle.com/javase/8/javase-clienttechnologies.htm)

* The new Modena theme has been implemented in this release. For more information, see the blog at [fxexperience.com](http://fxexperience.com/2013/03/modena-theme-update/).
* The new SwingNode class enables developers to embed Swing content into JavaFX applications. See the [SwingNode](http://docs.oracle.com/javase/8/javafx/api/javafx/embed/swing/SwingNode.html) javadoc and [Embedding Swing Content in JavaFX Applications](http://docs.oracle.com/javase/8/javafx/interoperability-tutorial/embed-swing.htm).
* The new UI Controls include the [DatePicker](http://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/DatePicker.html) and the [TreeTableView](http://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/TreeTableView.html) controls.
* The javafx.print package provides the public classes for the JavaFX Printing API. See the [javadoc](http://docs.oracle.com/javase/8/javafx/api/javafx/print/package-summary.html) for more information.
* The 3D Graphics features now include 3D shapes, camera, lights, subscene, material, picking, and antialiasing. The new Shape3D (Box, Cylinder, MeshView, and Spheresubclasses), SubScene, Material, PickResult, LightBase (AmbientLight andPointLight subclasses) , and SceneAntialiasing API classes have been added to the JavaFX 3D Graphics library. The Camera API class has also been updated in this release. See the corresponding class javadoc for javafx.scene.shape.Shape3D,javafx.scene.SubScene, javafx.scene.paint.Material,javafx.scene.input.PickResult, javafx.scene.SceneAntialiasing, and the[Getting Started with JavaFX 3D Graphics](http://docs.oracle.com/javase/8/javafx/graphics-tutorial/javafx-3d-graphics.htm) document.
* The WebView class provides new features and improvements. Review [Supported Features of HTML5](http://docs.oracle.com/javase/8/javafx/embedded-browser-tutorial/index.html) for more information about additional HTML5 features including Web Sockets, Web Workers, and Web Fonts.
* Enhanced text support including bi-directional text and complex text scripts such as Thai and Hindi in controls, and multi-line, multi-style text in text nodes.
* Support for Hi-DPI displays has been added in this release.
* The CSS Styleable\* classes became public API. See the [javafx.css](http://docs.oracle.com/javase/8/javafx/api/javafx/css/package-frame.html) javadoc for more information.
* The new [ScheduledService](http://docs.oracle.com/javase/8/javafx/api/javafx/concurrent/ScheduledService.html) class allows to automatically restart the service.
* JavaFX is now available for ARM platforms. JDK for ARM includes the base, graphics and controls components of JavaFX.

[Tools](http://docs.oracle.com/javase/8/docs/technotes/tools/enhancements-8.html)

* The jjs command is provided to invoke the Nashorn engine.
* The java command launches JavaFX applications.
* The java man page has been reworked.
* The jdeps command-line tool is provided for analyzing class files.
* Java Management Extensions (JMX) provide remote access to diagnostic commands.
* The jarsigner tool has an option for requesting a signed time stamp from a Time Stamping Authority (TSA).
* [Javac tool](http://docs.oracle.com/javase/8/docs/technotes/guides/javac/index.html)
* The -parameters option of the javac command can be used to store formal parameter names and enable the Reflection API to retrieve formal parameter names.
* The type rules for equality operators in the Java Language Specification (JLS) Section 15.21 are now correctly enforced by the javac command.
* The javac tool now has support for checking the content of javadoc comments for issues that could lead to various problems, such as invalid HTML or accessibility issues, in the files that are generated when javadoc is run. The feature is enabled by the new -Xdoclint option. For more details, see the output from running "javac -X". This feature is also available in the javadoc tool, and is enabled there by default.
* The javac tool now provides the ability to generate native headers, as needed. This removes the need to run the javah tool as a separate step in the build pipeline. The feature is enabled in javac by using the new -h option, which is used to specify a directory in which the header files should be written. Header files will be generated for any class which has either native methods, or constant fields annotated with a new annotation of type java.lang.annotation.Native.

[Javadoc tool](http://docs.oracle.com/javase/8/docs/technotes/guides/javadoc/whatsnew-8.html)

* The javadoc tool supports the new DocTree API that enables you to traverse Javadoc comments as abstract syntax trees.
* The javadoc tool supports the new Javadoc Access API that enables you to invoke the Javadoc tool directly from a Java application, without executing a new process. See the[javadoc what's new](http://docs.oracle.com/javase/8/docs/technotes/guides/javadoc/whatsnew-8.html) page for more information.
* The javadoc tool now has support for checking the content of javadoc comments for issues that could lead to various problems, such as invalid HTML or accessibility issues, in the files that are generated when javadoc is run. The feature is enabled by default, and can also be controlled by the new -Xdoclint option. For more details, see the output from running "javadoc -X". This feature is also available in the javac tool, although it is not enabled by default there.
* [Internationalization](http://docs.oracle.com/javase/8/docs/technotes/guides/intl/enhancements.8.html)
* Unicode Enhancements, including support for Unicode 6.2.0
* Adoption of Unicode CLDR Data and the java.locale.providers System Property
* New Calendar and Locale APIs
* Ability to Install a Custom Resource Bundle as an Extension

[Deployment](http://docs.oracle.com/javase/8/docs/technotes/guides/jweb/enhancements-8.html)

* For sandbox applets and Java Web Start applications, URLPermission is now used to allow connections back to the server from which they were started. SocketPermission is no longer granted.
* The Permissions attribute is required in the JAR file manifest of the main JAR file at all security levels.

[Date-Time Package](http://docs.oracle.com/javase/8/docs/technotes/guides/datetime/index.html) - a new set of packages that provide a comprehensive date-time model.

[Scripting](http://docs.oracle.com/javase/8/docs/technotes/guides/scripting/enhancements.html#jdk8)

* [Nashorn](http://docs.oracle.com/javase/8/docs/technotes/guides/scripting/nashorn/) Javascript Engine

[Pack200](http://docs.oracle.com/javase/8/docs/technotes/guides/pack200/enhancements.html)

* Pack200 Support for Constant Pool Entries and New Bytecodes Introduced by JSR 292
* JDK8 support for class files changes specified by JSR-292, JSR-308 and JSR-335

[IO and NIO](http://docs.oracle.com/javase/8/docs/technotes/guides/io/enhancements.html#jdk8)

* New SelectorProvider implementation for Solaris based on the Solaris event port mechanism. To use, run with the system property java.nio.channels.spi.Selectorset to the value sun.nio.ch.EventPortSelectorProvider.
* Decrease in the size of the <JDK\_HOME>/jre/lib/charsets.jar file
* Performance improvement for the java.lang.String(byte[], \*) constructor and thejava.lang.String.getBytes() method.
* [java.lang and java.util Packages](http://docs.oracle.com/javase/8/docs/technotes/guides/lang/enhancements.html#jdk8)
* Parallel Array Sorting
* Standard Encoding and Decoding Base64
* Unsigned Arithmetic Support

[JDBC](http://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/)

* The JDBC-ODBC Bridge has been removed.
* JDBC 4.2 introduces new features.
* Java DB
* JDK 8 includes Java DB 10.10.

[Networking](http://docs.oracle.com/javase/8/docs/technotes/guides/net/enhancements-8.0.html)

* The class java.net.URLPermission has been added.
* In the class java.net.HttpURLConnection, if a security manager is installed, calls that request to open a connection require permission.
* [Concurrency](http://docs.oracle.com/javase/8/docs/technotes/guides/concurrency/changes8.html)
* Classes and interfaces have been added to the java.util.concurrent package.
* Methods have been added to the java.util.concurrent.ConcurrentHashMap class to support aggregate operations based on the newly added streams facility and lambda expressions.
* Classes have been added to the java.util.concurrent.atomic package to support scalable updatable variables.
* Methods have been added to the java.util.concurrent.ForkJoinPool class to support a common pool.
* The java.util.concurrent.locks.StampedLock class has been added to provide a capability-based lock with three modes for controlling read/write access.

[Java XML](http://docs.oracle.com/javase/8/docs/technotes/guides/xml/enhancements.html) - [JAXP](http://docs.oracle.com/javase/8/docs/technotes/guides/xml/jaxp/enhancements-8.html)

[HotSpot](http://docs.oracle.com/javase/8/docs/technotes/guides/vm/)

* Hardware intrinsics were added to use Advanced Encryption Standard (AES). The UseAESand UseAESIntrinsics flags are available to enable the hardware-based AES intrinsics for Intel hardware. The hardware must be 2010 or newer Westmere hardware. For example, to enable hardware AES, use the following flags:
* -XX:+UseAES -XX:+UseAESIntrinsics
* To disable hardware AES use the following flags:
* -XX:-UseAES -XX:-UseAESIntrinsics
* Removal of PermGen.
* Default Methods in the Java Programming Language are supported by the byte code instructions for method invocation.

**Micro Services Architecture**

* Understanding the Micro Services Architecture
* Best practices & guidelines
* Spring boot vs Play Framework
* Spring Boot Architectural deep dive

**Conclusion**

* Live Working Sample following the micro-services architecture via. Spring boot over Java 8 features