Advanced Java

Course Topics

Simon Roberts, January 2016

Advanced OO Features

- Meanings of final
  - Designed for inheritance?
  - Creating constants
- Meaning of equality in OO
- More enum features
  - Overriding toString() vs. name()
  - Arbitrary methods and fields
  - Constructors
- Inner classes
  - static (builder)
  - instance (iterator)
  - Anonymous inner, closure & effectively final

Advanced Exception Handling

- Design with exceptions
  - Design perspective of try catch and finally
  - Maintaining encapsulation
  - Designing the API
- Multi-catch
  - Rules for, and type of, catch formal parameters
  - Rethrowing, the “cause”
- Suppressed exceptions finally

Generics

- Defining a generic class / interface
- Type erasure and non-reifiable types
- Bounded generic types <E extends Xxx>
- Multiple bounds <E extends Aclass & AnIf & OtherIf>
- Defining a generic method
- Upper-, lower-, and un-bounded wildcard types

Dynamic Java

- Annotations
  - Nature
  - Retention / RetentionPolicy

    - Target / ElementType
- Runtime annotation processing with reflection
    - Dynamic class loading, instantiation
    - Finding methods, fields
    - Finding annotations
    - Invoking methods
    - Accessing fields
    - Bypassing restrictions of private
- Annotations with fields
    - Valid types
    - Field names and the special name “value”
- Default values

Functional Java

- Basic Functional programming concepts
    - Behavior as arguments and return values
    - Passing behavior in: generic sub-list creation method
    - Returning behavior: combining filters
- Lambda expressions
- Standard functional interfaces in java.util.function

Java 8 Enhancements

- static methods
- default methods
- Method references
- Collections API enhancements

Streams

- Stream concept
- Terminal vs non-terminal operations
- forEach
- filter
- map
- flatMap
- reduce and collect
- Collectors
- Primitive streams, boxing / unboxing
- More ways of obtaining Streams:

continued...

    - Streams factories: empty, generate, iterate, of
    - StreamSupport, Spliterator and Iterable
    - Files class methods
- Closing streams
- Parallel, Sequential, Ordered, Unordered, concurrency and hidden impediments/costs

Threading

- Thread and Runnable
- Thread cooperation
    - Data corruption
    - Overview of Java's memory model, happens-before relationships
    - Overview of synchronized, wait, notify, & volatile, also Semaphore
    - ReentrantLock
- Pipeline architectures, BlockingQueue
    - Benefits of immutability
- ExecutorService, Callable, Future and job control
- Synchronizers: join, CyclicBarrier, CountDownLatch
- Concurrrent collections
- Concurrency and scalability
    - Amdahl's law
    - Atomics and Accumulators
    - CopyOnWrite structures
    - ThreadLocalRandom
- ThreadLocal variables
- CompletableFuture API