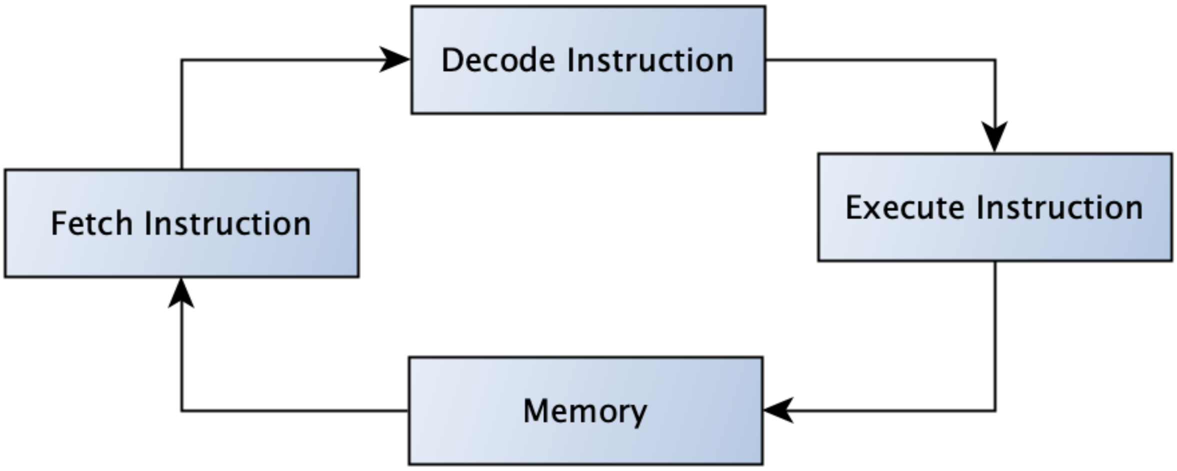
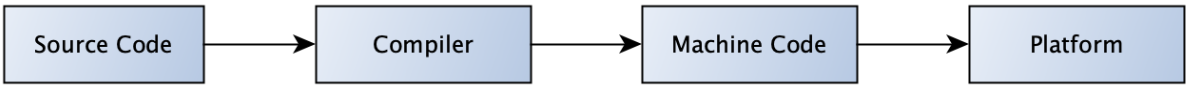
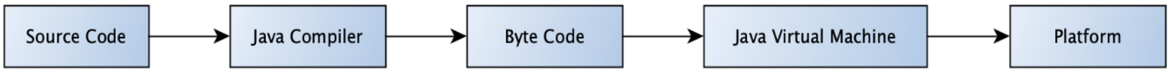
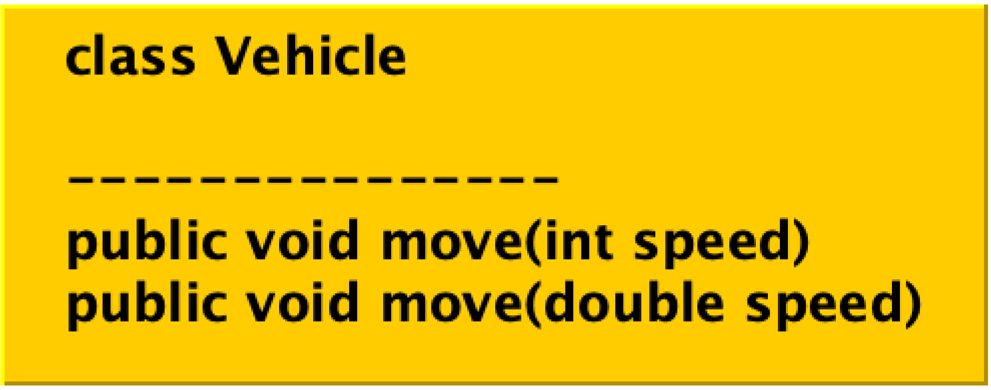
Chapter 1: Getting Into Software Design Patterns

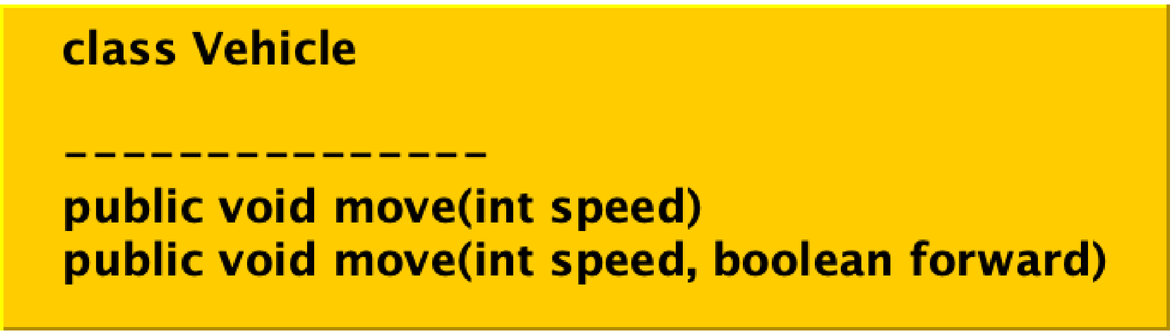




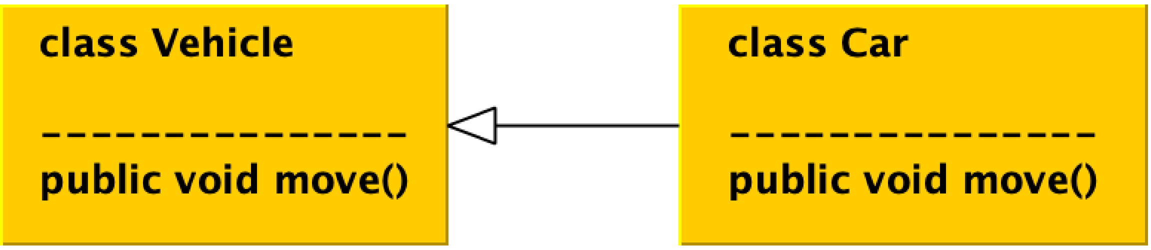




```
class Vehicle

----------------
public void move(int speed)
public void move(double speed)
```

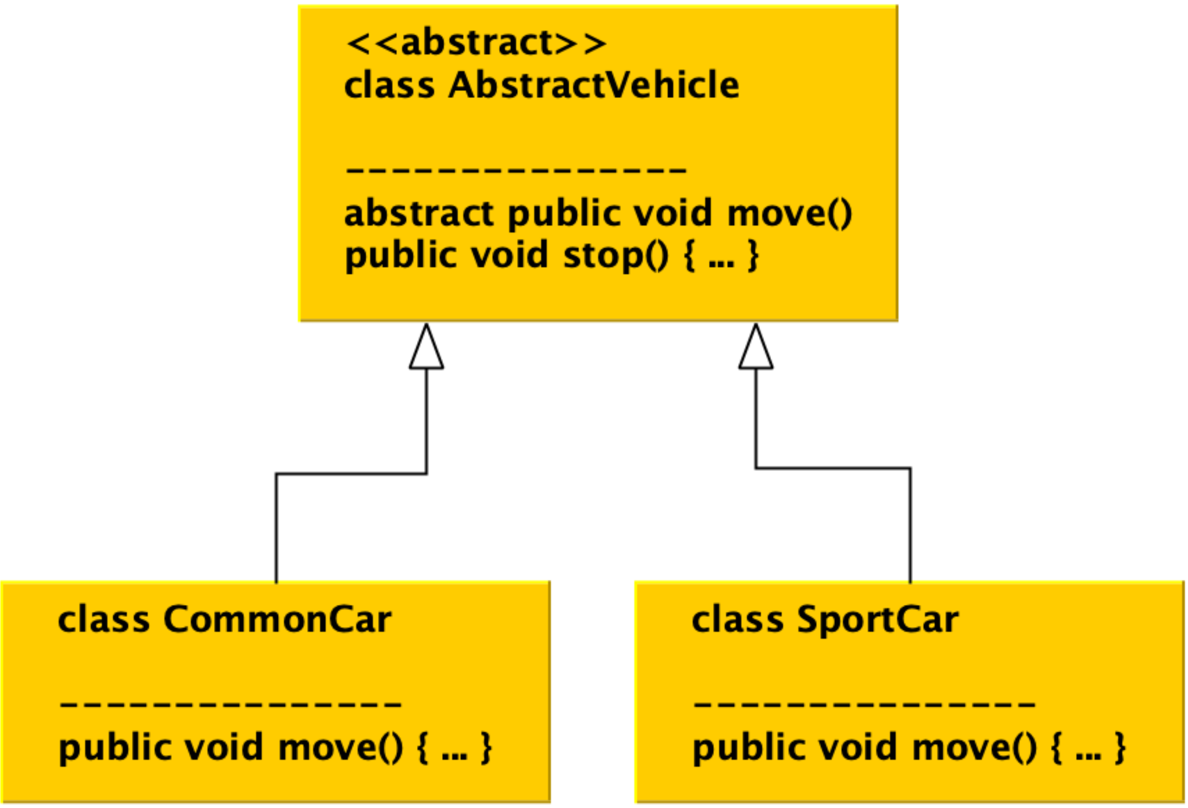


```
class Vehicle

---------------
public void move(int speed)
public void move(int speed, boolean forward)
```



```
class Vehicle

---------------
public void move()
```

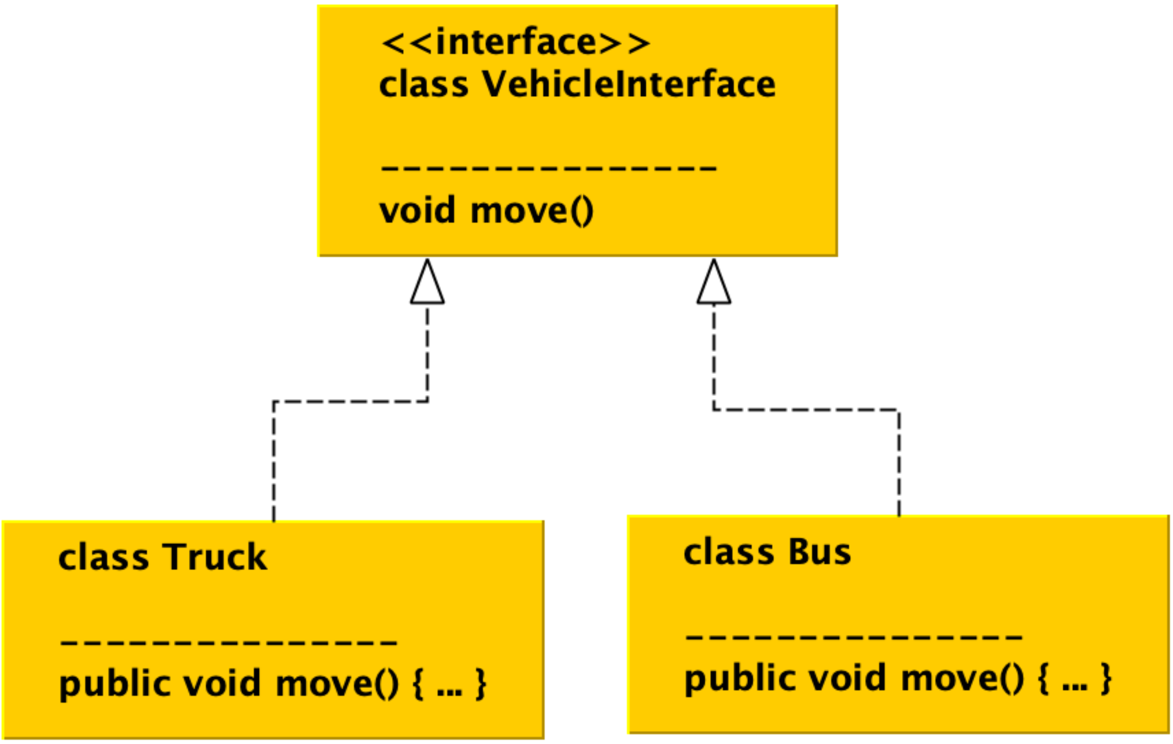
```
class Car

---------------
public void move()
```

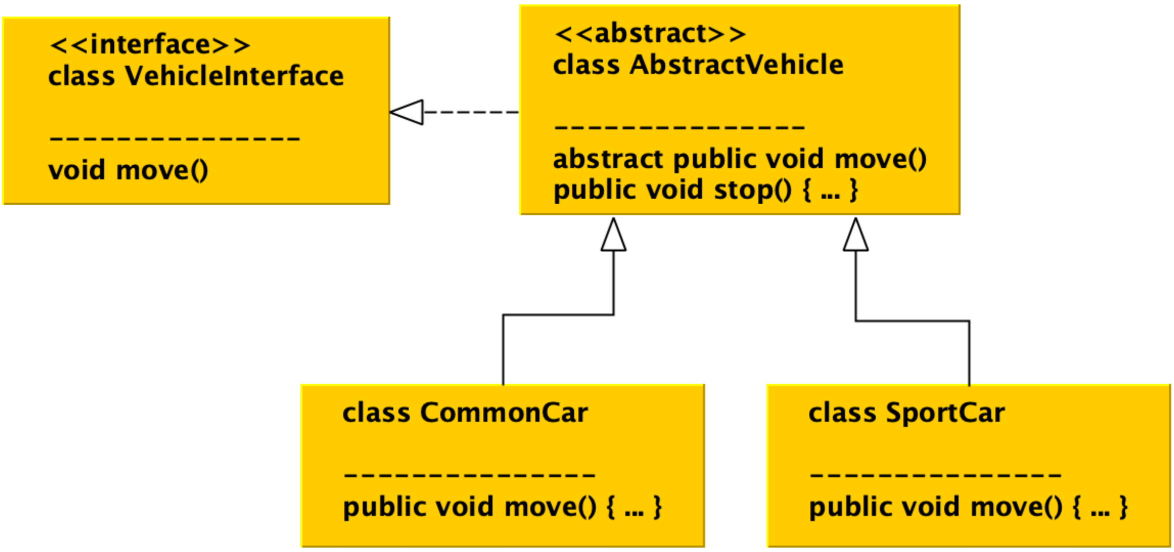


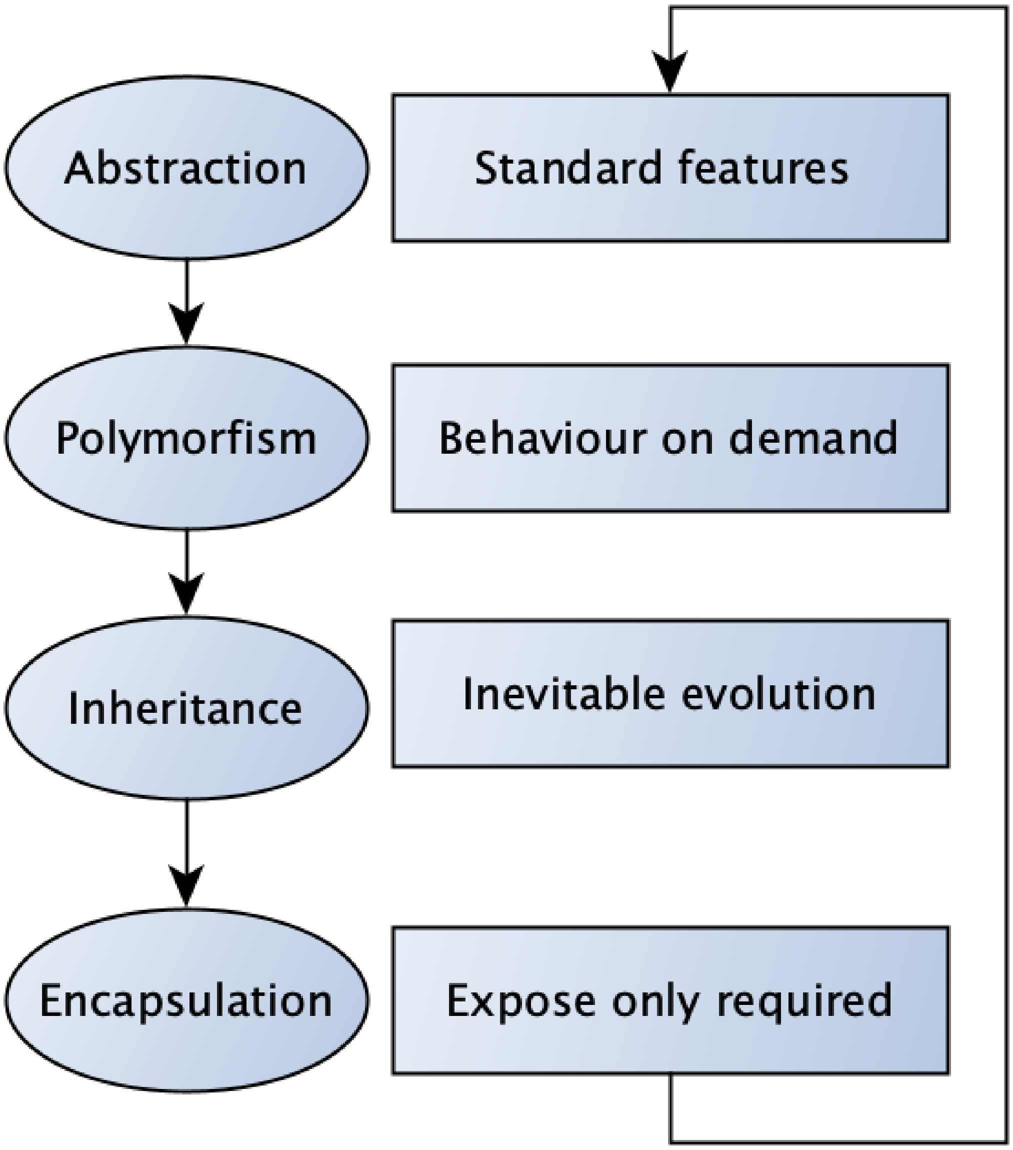
```
<<abstract>>
class AbstractVehicle

---------------
abstract public void move()
public void stop() { ... }
```

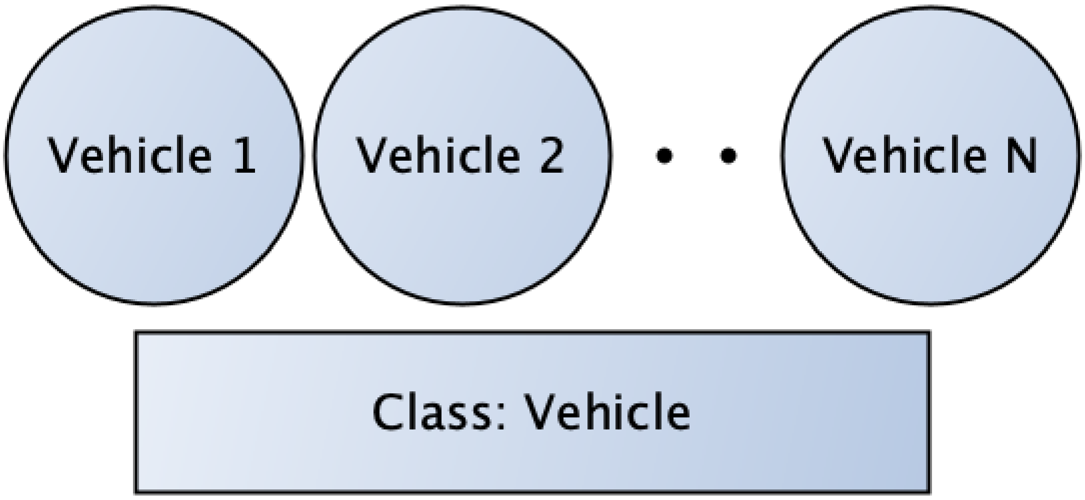
```
class CommonCar

---------------
public void move() { ... }
```

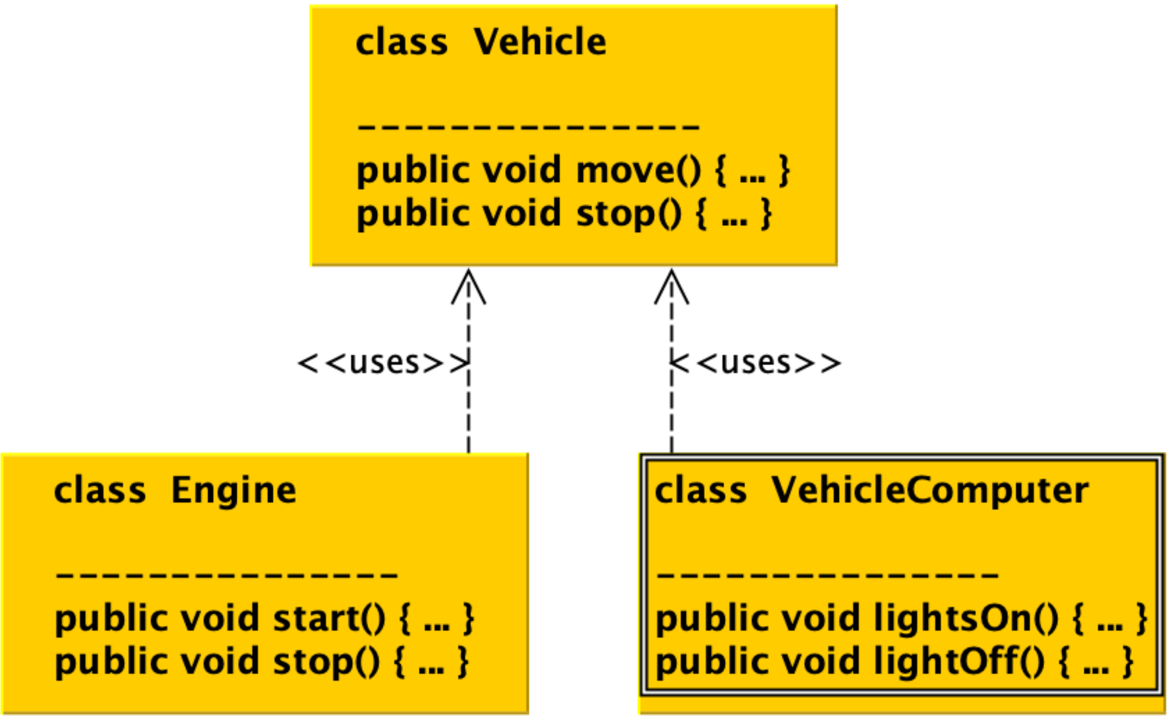
```
class SportCar

---------------
public void move() { ... }
```


<<interface>>
class VehicleInterface
----
void move()
class Truck
----
public void move() { ... }
class Bus
----
public void move() { ... }

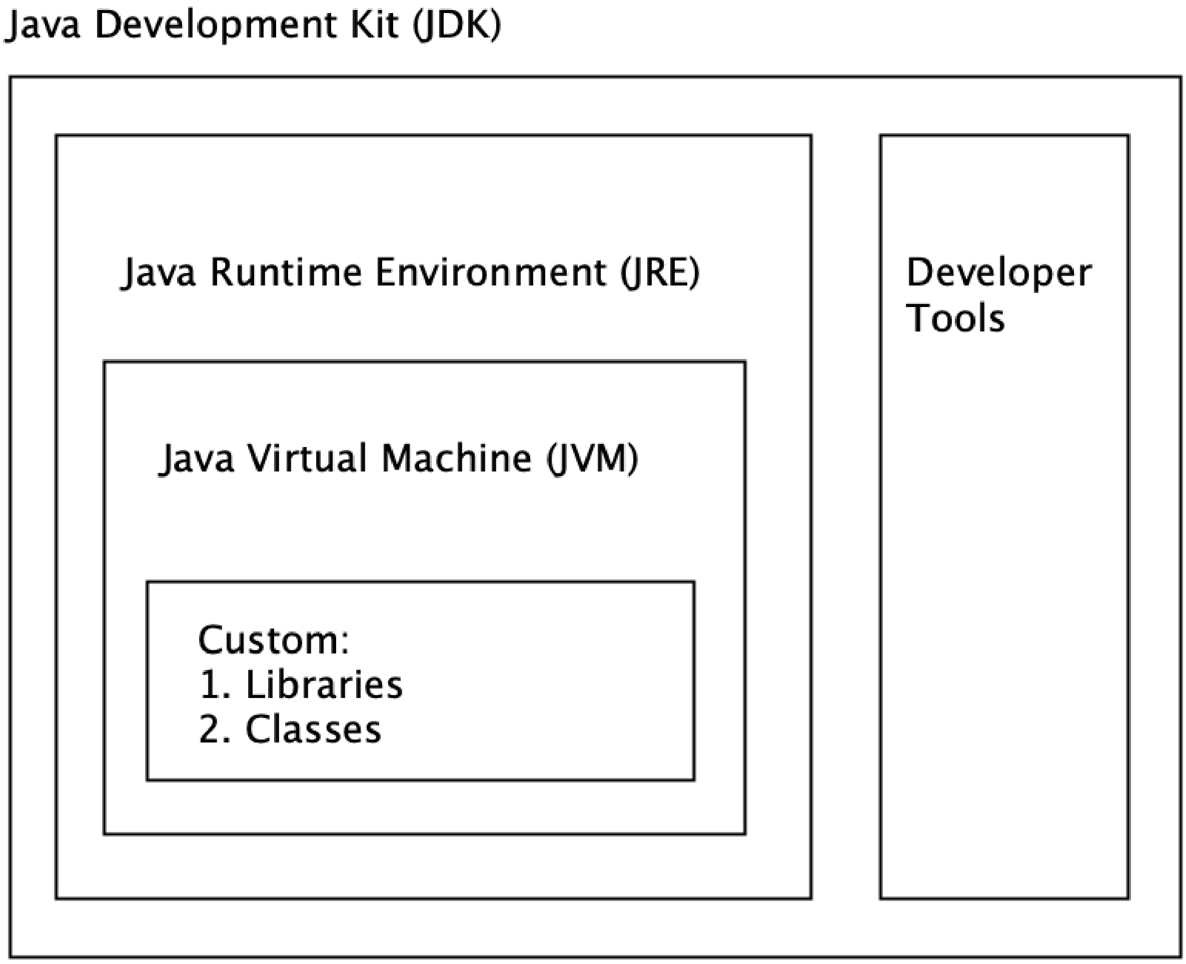

<<interface>>
class VehicleInterface
----
void move()
<<abstract>>
class AbstractVehicle
----
abstract public void move()
public void stop() { ... }
class CommonCar
----
public void move() { ... }
class SportCar
----
public void move() { ... }

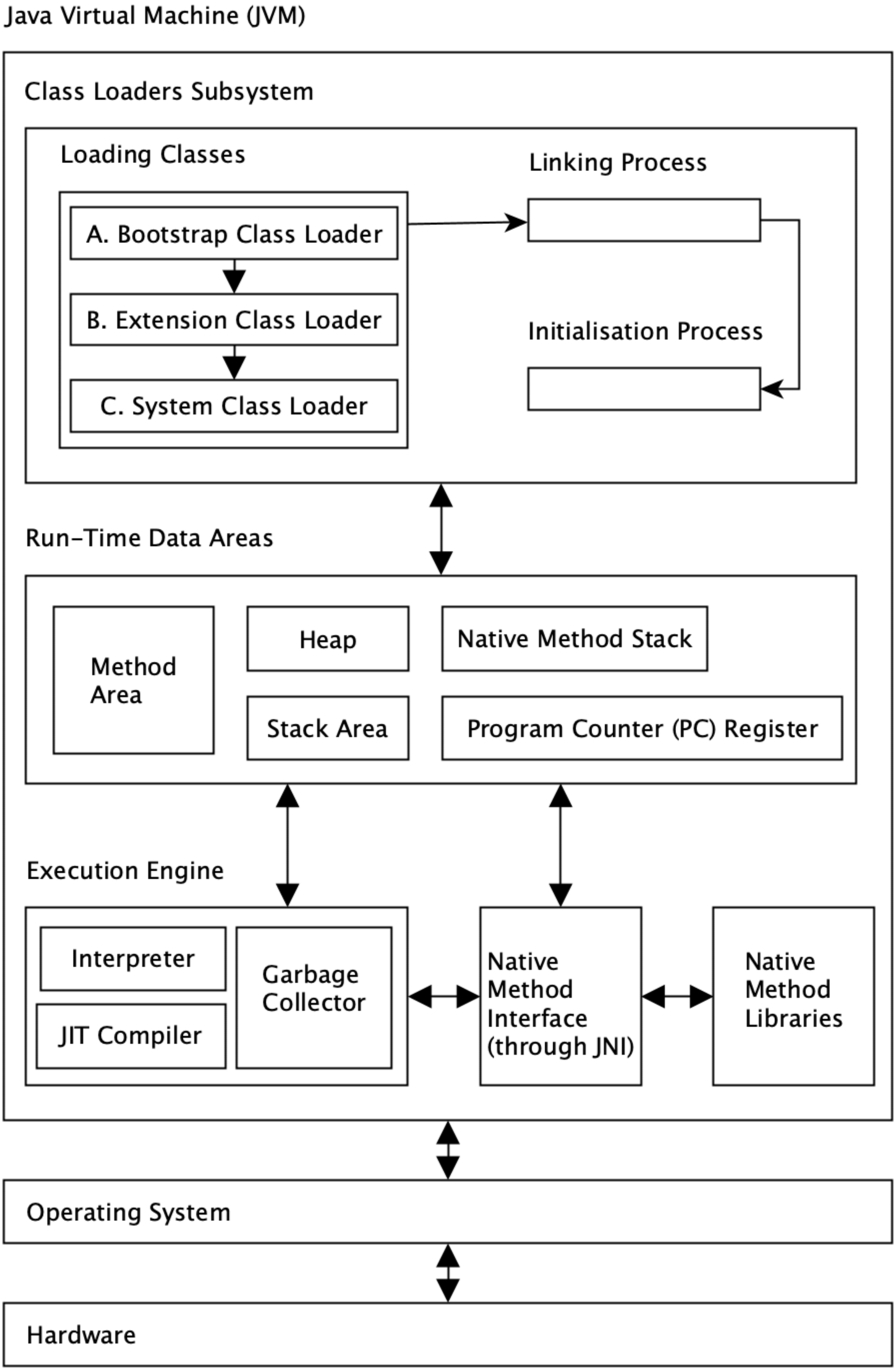

Abstraction
Standard features
Polymorfism
Behaviour on demand
Inheritance
Inevitable evolution
Encapsulation
Expose only required

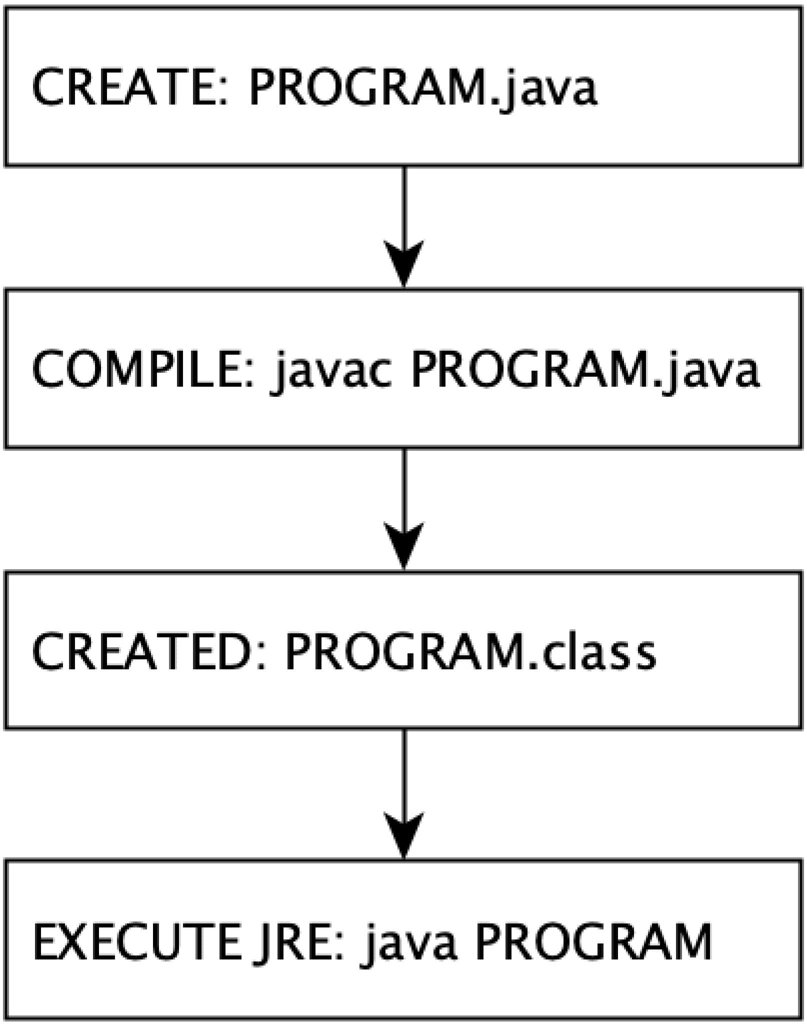

Vehicle 1
Vehicle 2
Vehicle N
Class: Vehicle

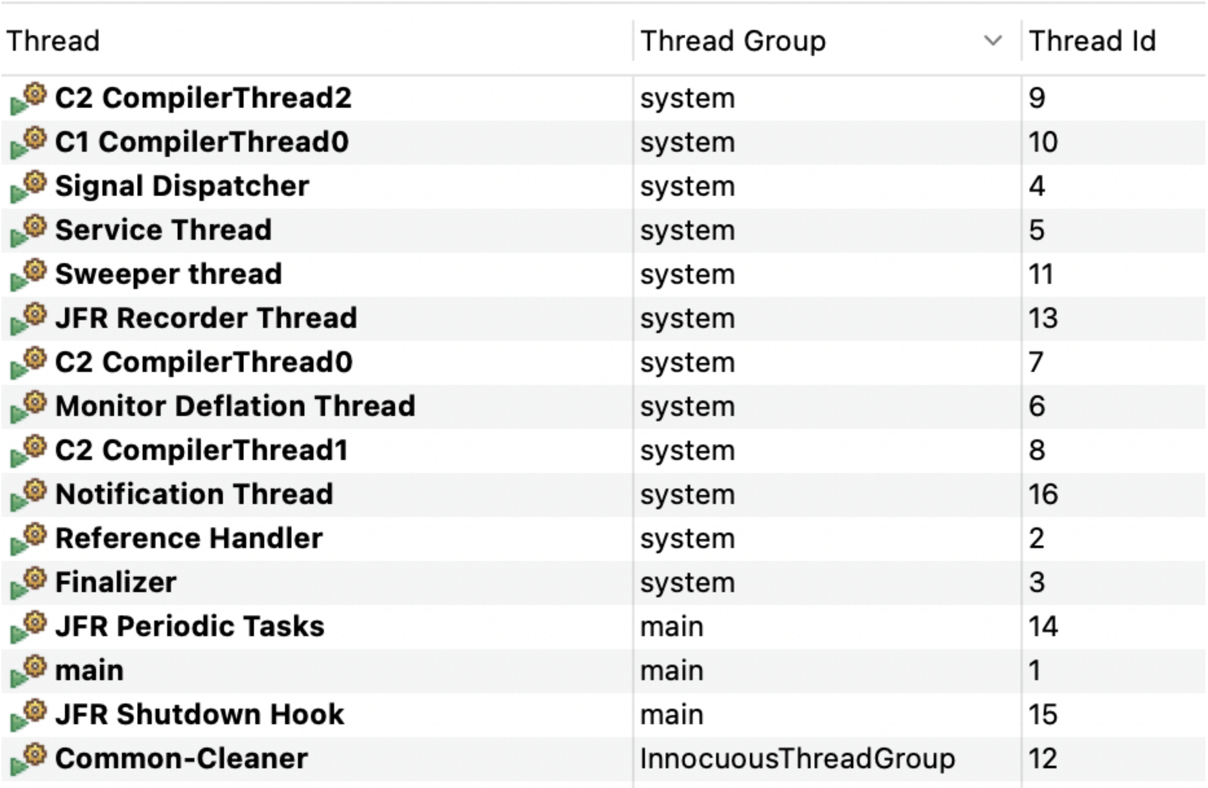

class Vehicle
----
public void move() { ... }
public void stop() { ... }
<<uses>>
<<uses>>
class Engine
----
public void start() { ... }
public void stop() { ... }
class VehicleComputer
----
public void lightsOn() { ... }
public void lightOff() { ... }

Chapter 2: Discovering the Java Platform for Design Patterns

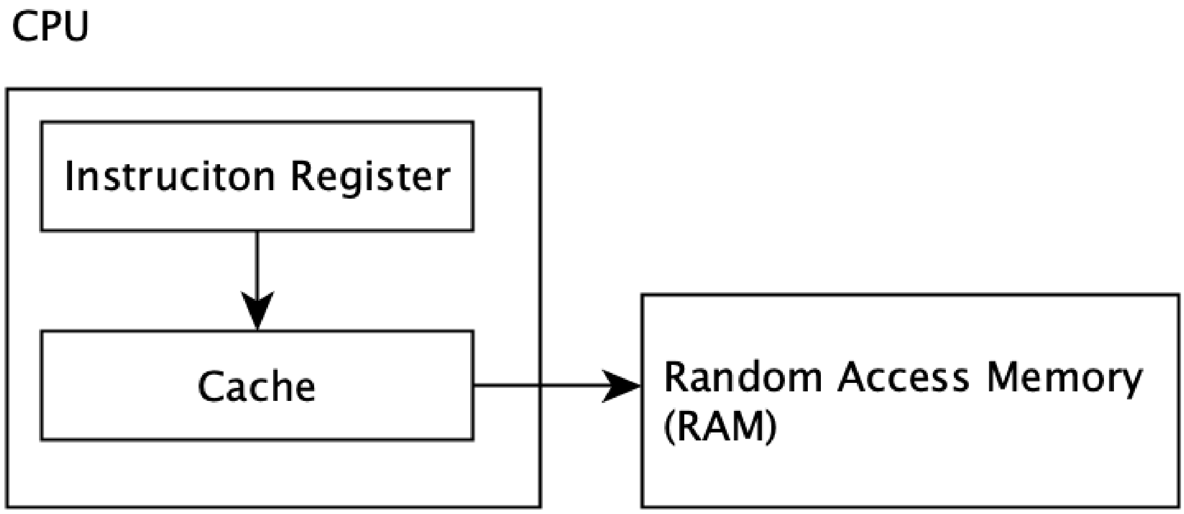

Java Development Kit (JDK)
Java Runtime Environment (JRE)
Java Virtual Machine (JVM)
Custom:
1. Libraries
2. Classes
Developer
Tools


Java Virtual Machine (JVM)
Class Loaders Subsystem
Loading Classes
A. Bootstrap Class Loader
B. Extension Class Loader
C. System Class Loader
Linking Process
Initialisation Process
Run-Time Data Areas
Method Area
Heap
Native Method Stack
Stack Area
Program Counter (PC) Register
Execution Engine
Interpreter
JIT Compiler
Garbage Collector
Native Method Interface (through JNI)
Native Method Libraries
Operating System
Hardware

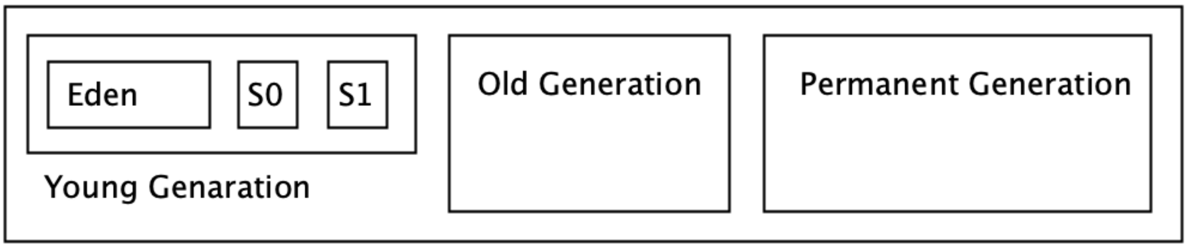

CREATE: PROGRAM.java
COMPILE: javac PROGRAM.java
CREATED: PROGRAM.class
EXECUTE JRE: java PROGRAM



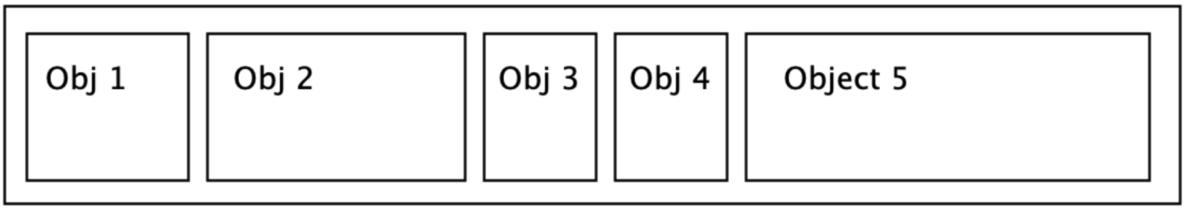
Thread	Thread Group	Thread Id
C2 CompilerThread2	system	9
C1 CompilerThread0	system	10
Signal Dispatcher	system	4
Service Thread	system	5
Sweeper thread	system	11
JFR Recorder Thread	system	13
C2 CompilerThread0	system	7
Monitor Deflation Thread	system	6
C2 CompilerThread1	system	8
Notification Thread	system	16
Reference Handler	system	2
Finalizer	system	3
JFR Periodic Tasks	main	14
main	main	1
JFR Shutdown Hook	main	15
Common-Cleaner	InnocuousThreadGroup	12

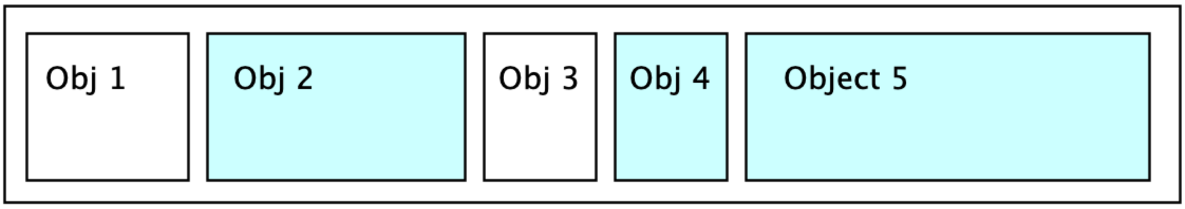


Heap strucutre - hotspot



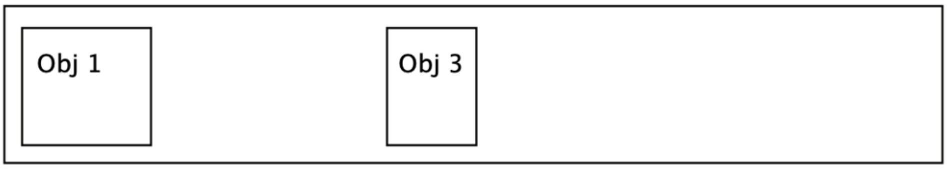
Allocated Object before Marking



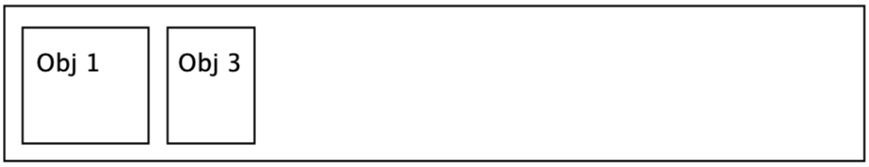


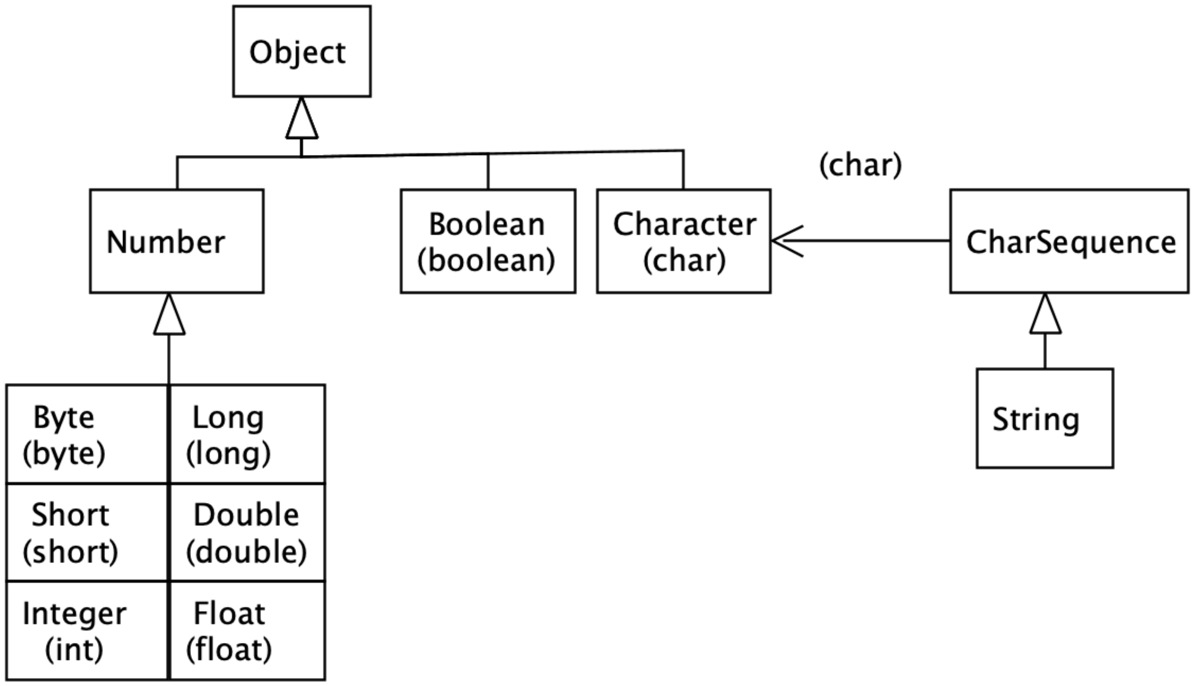
Allocated Object after Marking

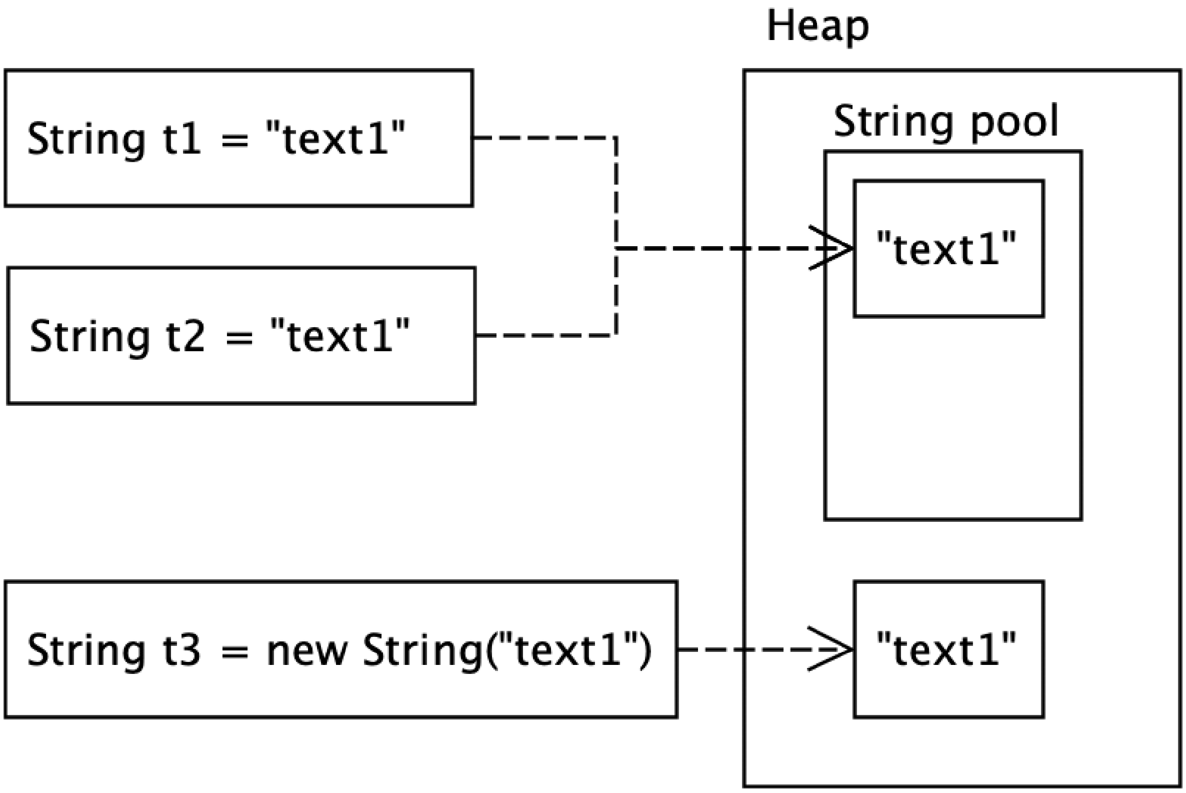
Standard deletion step

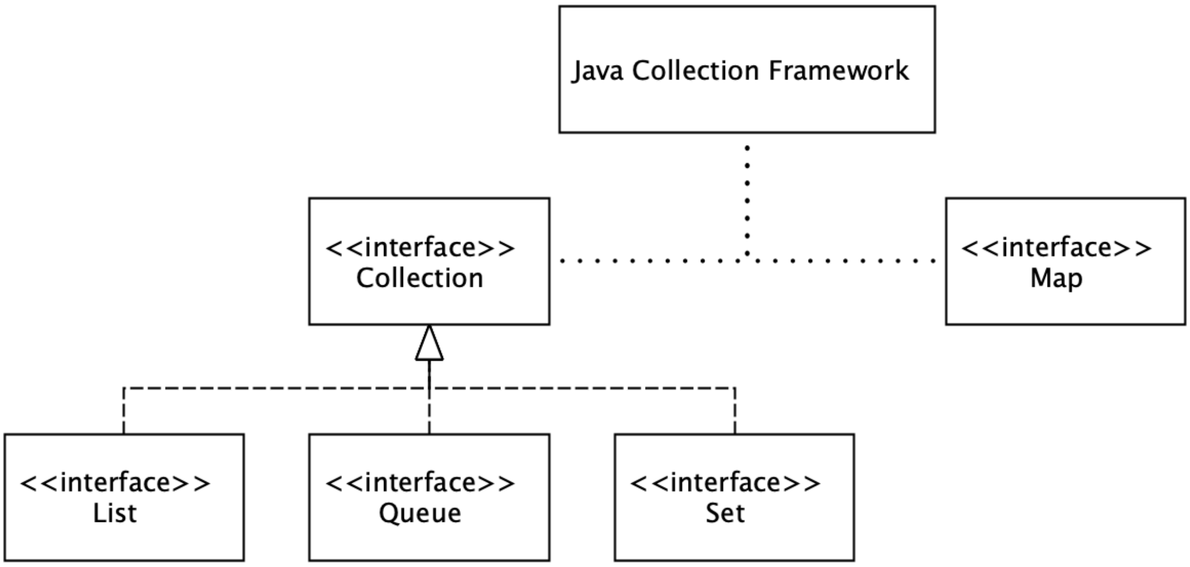


Compacting step

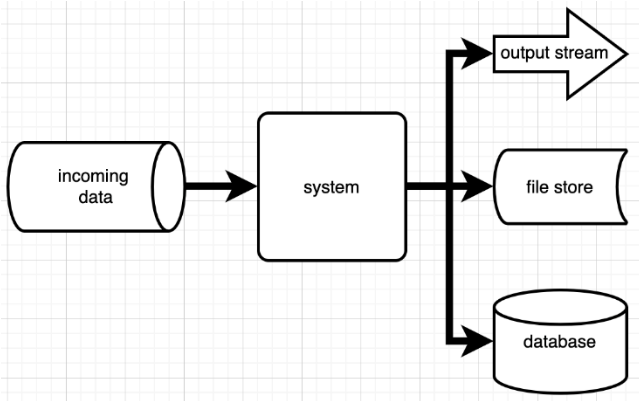


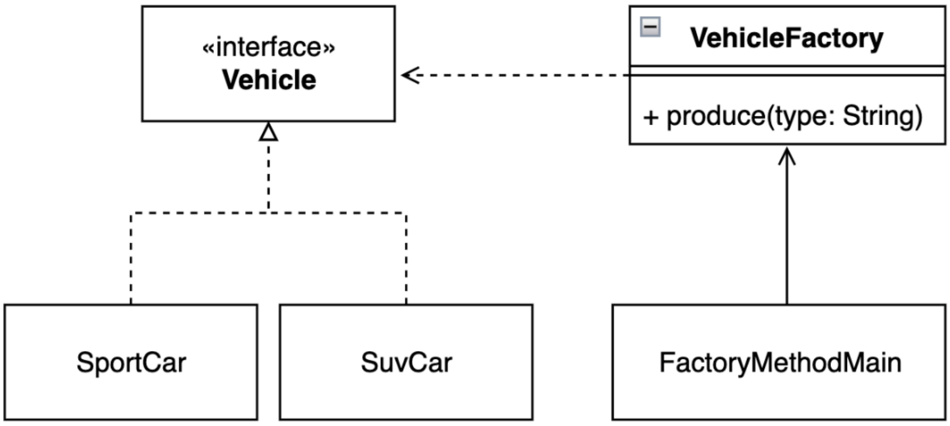

Object
Number
Boolean
(boolean)
Character
(char)
(char)
CharSequence
String
Byte
(byte)
Long
(long)
Short
(short)
Double
(double)
Integer
(int)
Float
(float)

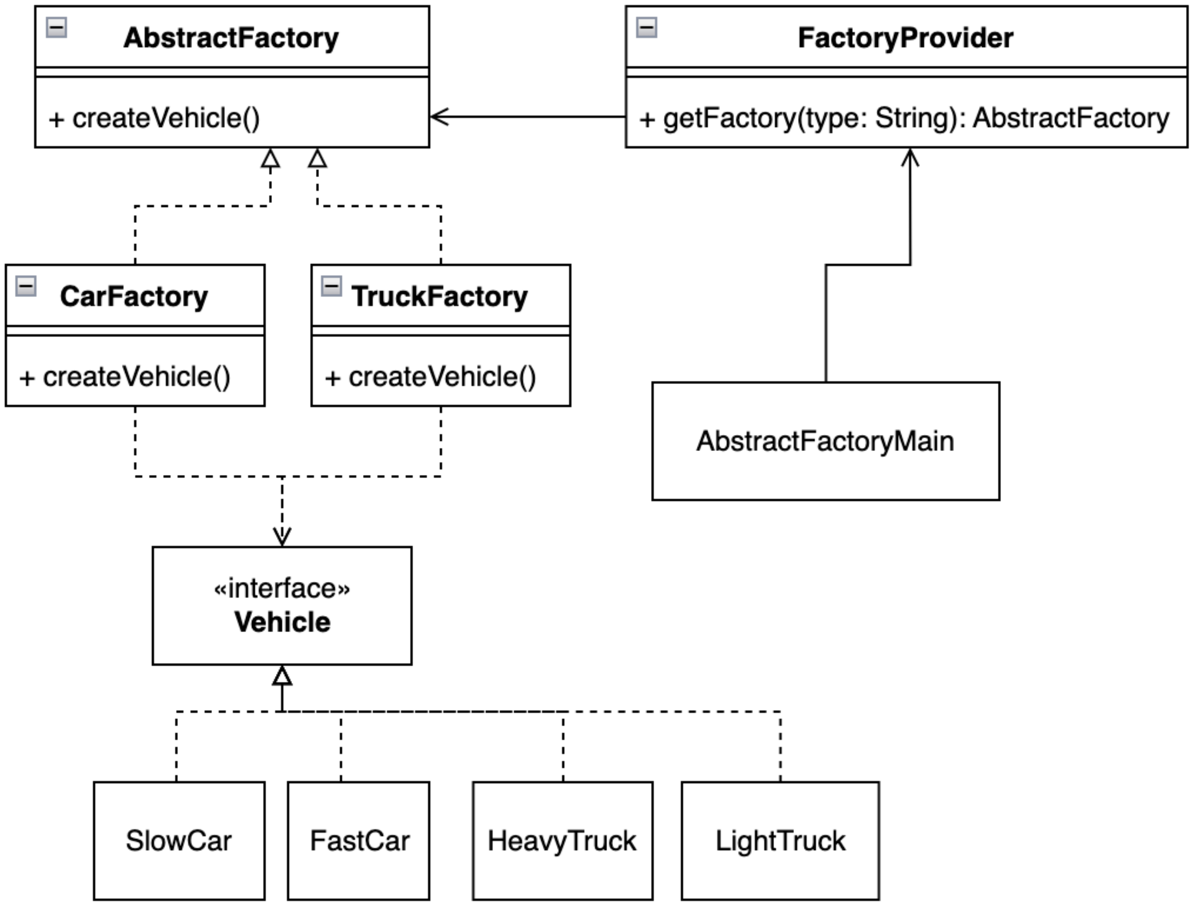

Heap
String pool
String t1 = "text1"
String t2 = "text1"
"text1"
String t3 = new String("text1")
"text1"

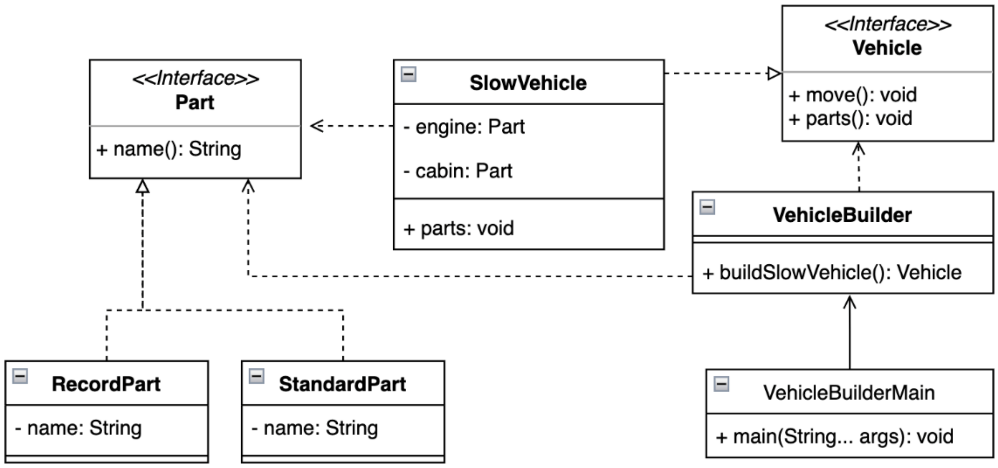

Java Collection Framework
<<interface>> Collection
<<interface>> Map
<<interface>> List
<<interface>> Queue
<<interface>> Set

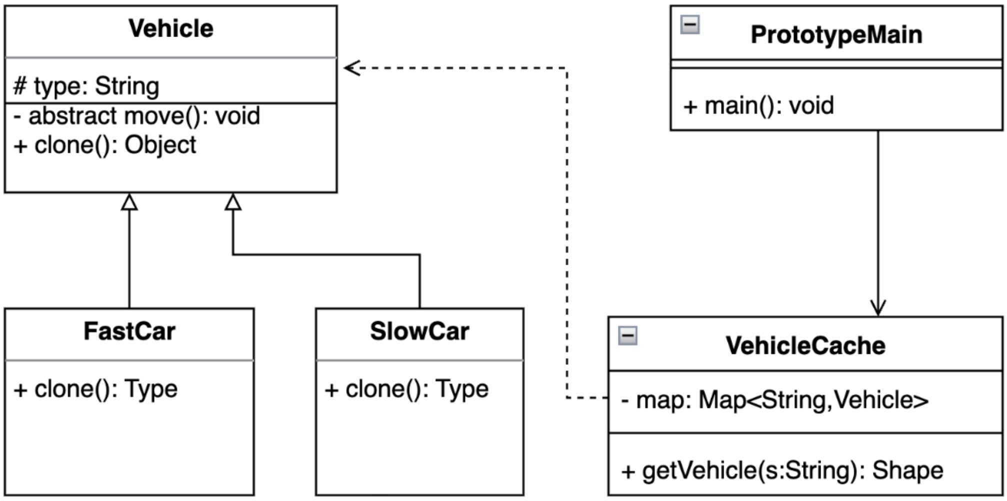
Chapter 3: Working with Creational Design Patterns

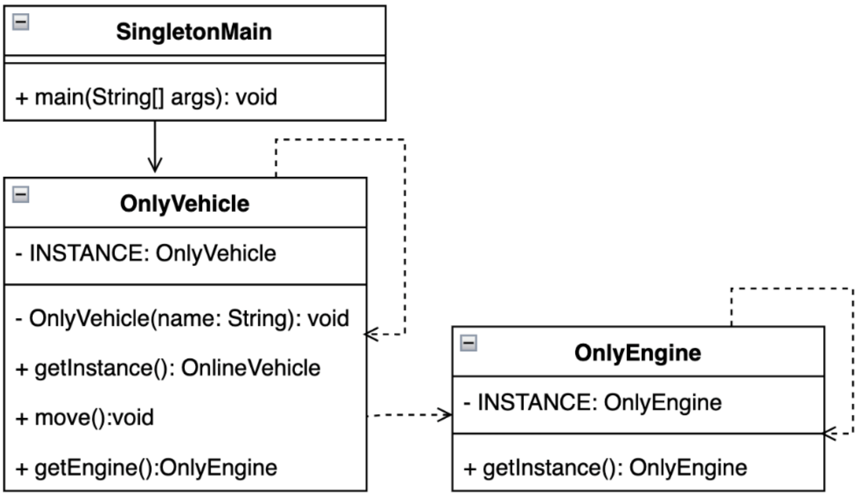


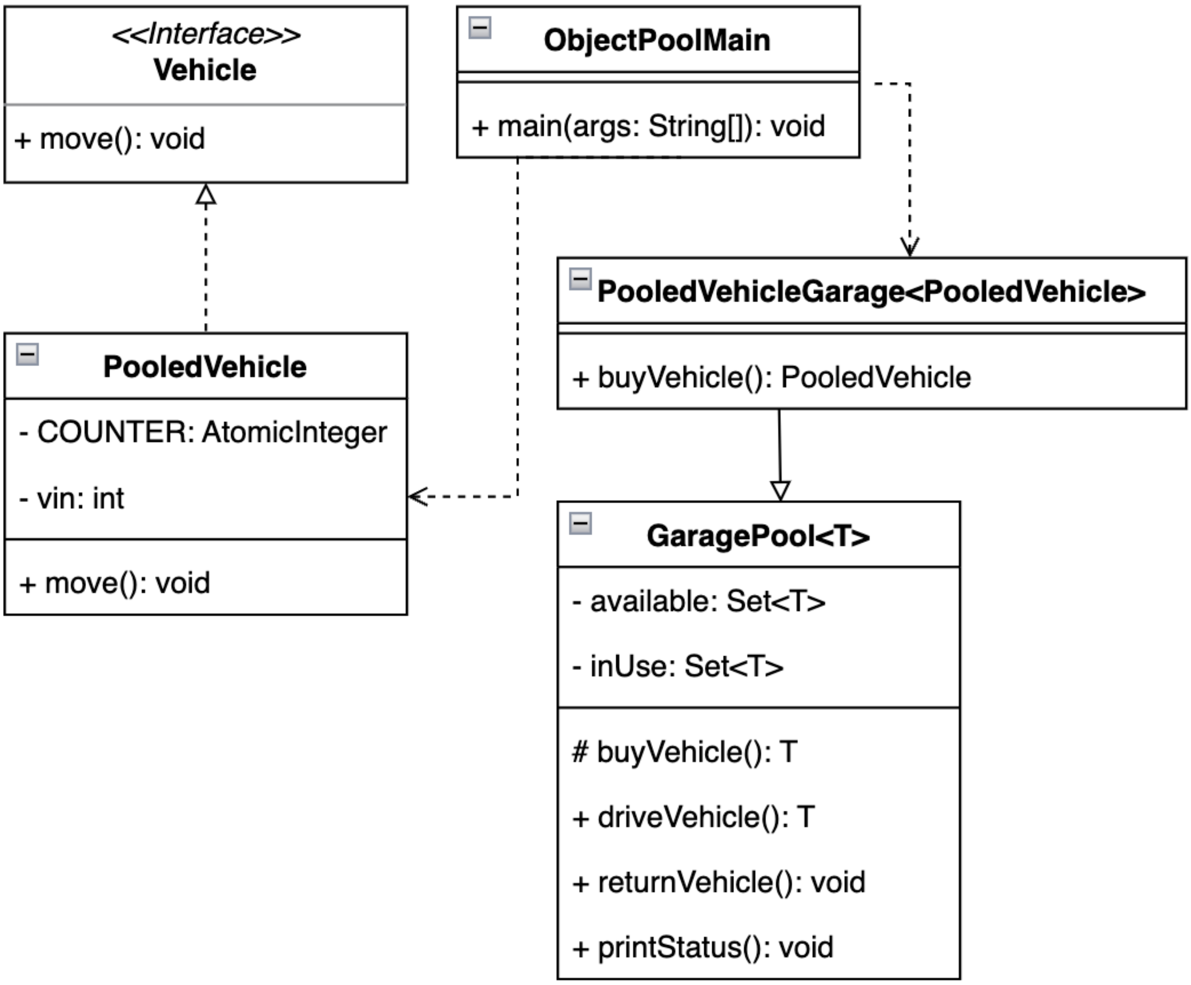



AbstractFactory
+ createVehicle()
FactoryProvider
+ getFactory(type: String): AbstractFactory
CarFactory
+ createVehicle()
TruckFactory
+ createVehicle()
AbstractFactoryMain
«interface»
Vehicle
SlowCar
FastCar
HeavyTruck
LightTruck

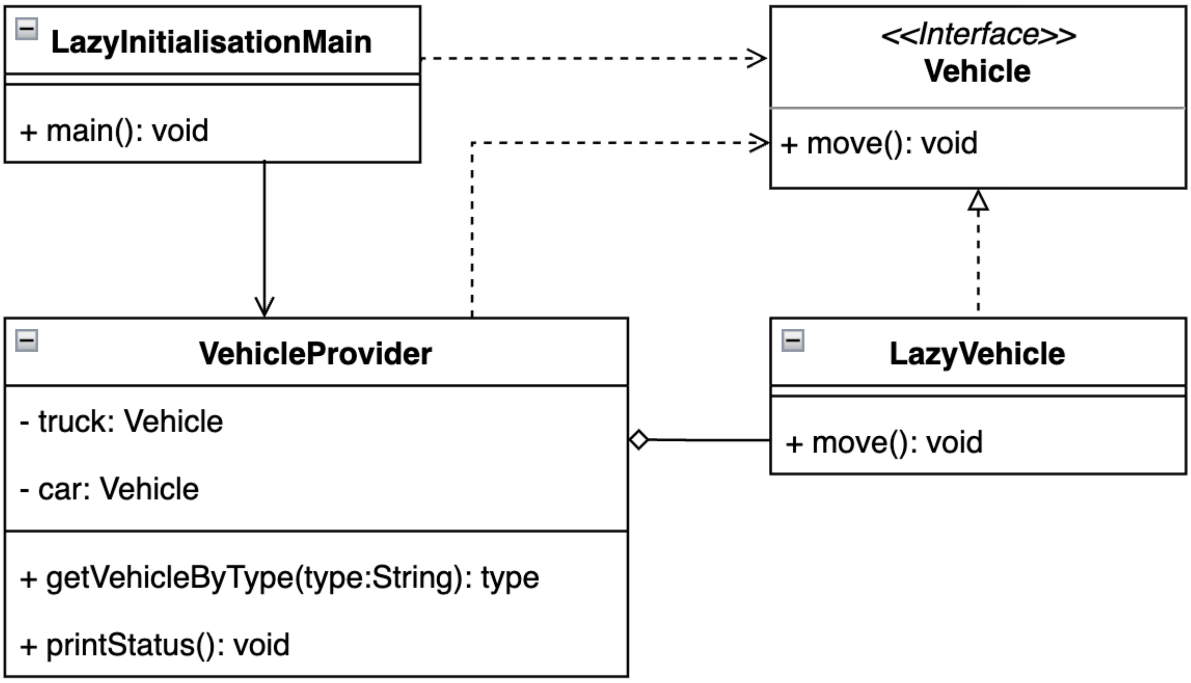

<<Interface>>
Part
+ name(): String
SlowVehicle
- engine: Part
- cabin: Part
+ parts: void
<<Interface>>
Vehicle
+ move(): void
+ parts(): void
VehicleBuilder
+ buildSlowVehicle(): Vehicle
RecordPart
- name: String
StandardPart
- name: String
VehicleBuilderMain
+ main(String... args): void


Vehicle
# type: String
- abstract move(): void
+ clone(): Object
FastCar
+ clone(): Type
SlowCar
+ clone(): Type
PrototypeMain
+ main(): void
VehicleCache
- map: Map<String,Vehicle>
+ getVehicle(s:String): Shape

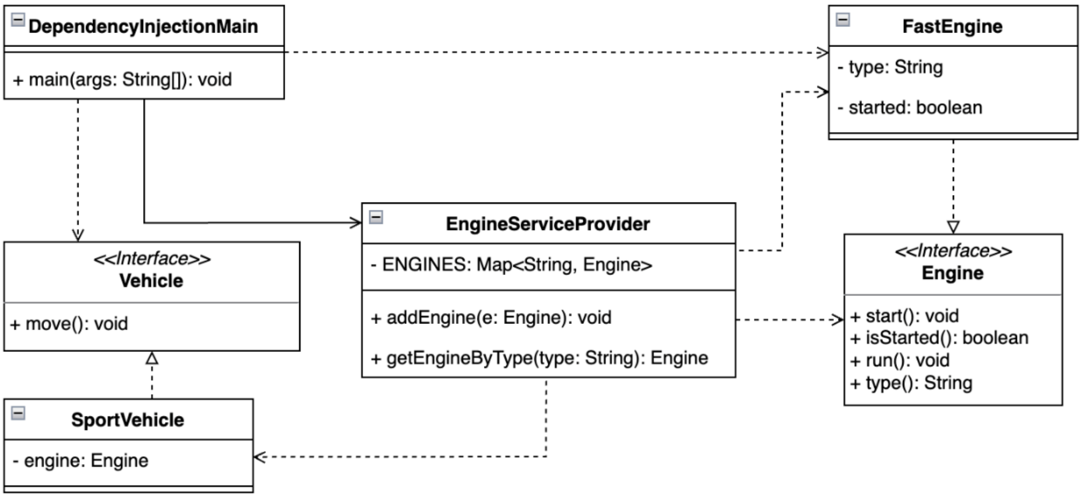

SingletonMain
+ main(String[] args): void
OnlyVehicle
- INSTANCE: OnlyVehicle
- OnlyVehicle(name: String): void
+ getInstance(): OnlineVehicle
+ move():void
+ getEngine():OnlyEngine
OnlyEngine
- INSTANCE: OnlyEngine
+ getInstance(): OnlyEngine



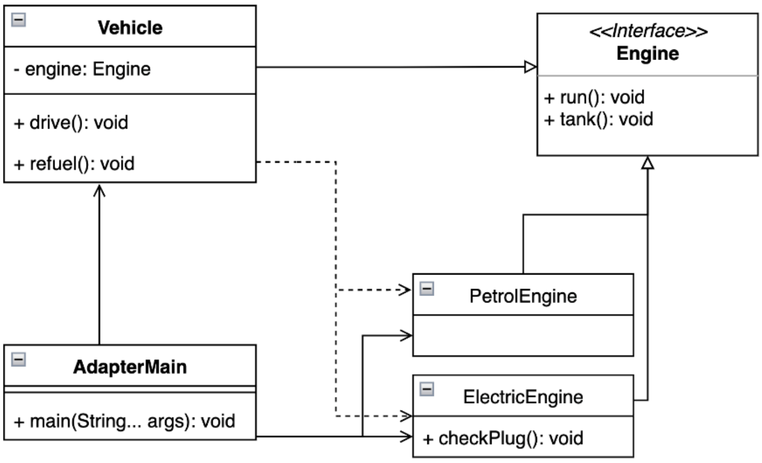
<<Interface>>
Vehicle
+ move(): void
ObjectPoolMain
+ main(args: String[]): void
PooledVehicleGarage<PooledVehicle>
+ buyVehicle(): PooledVehicle
PooledVehicle
- COUNTER: AtomicInteger
- vin: int
+ move(): void
GaragePool<T>
- available: Set<T>
- inUse: Set<T>
# buyVehicle(): T
+ driveVehicle(): T
+ returnVehicle(): void
+ printStatus(): void

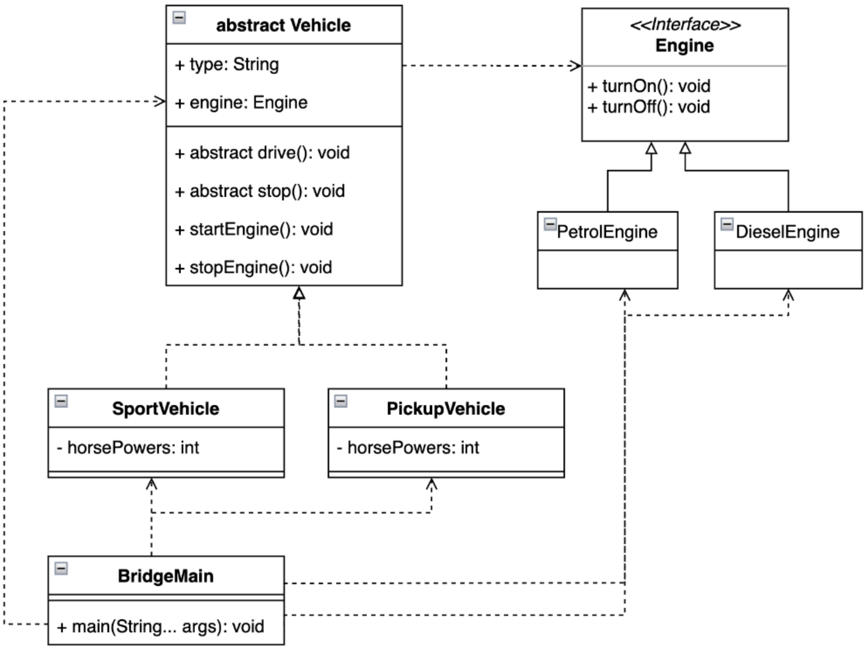


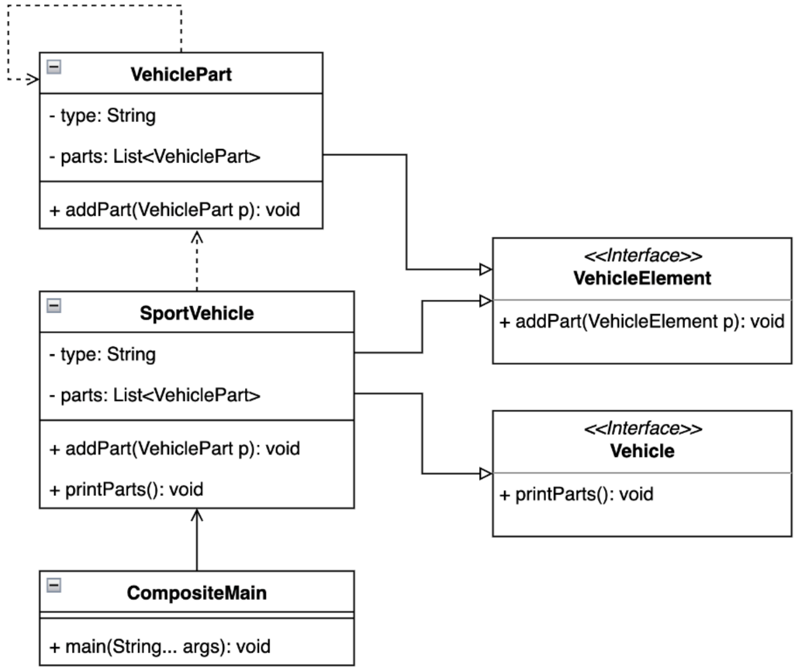
LazyInitialisationMain
+ main(): void
<<Interface>>
Vehicle
+ move(): void
VehicleProvider
- truck: Vehicle
- car: Vehicle
+ getVehicleByType(type:String): type
+ printStatus(): void
LazyVehicle
+ move(): void

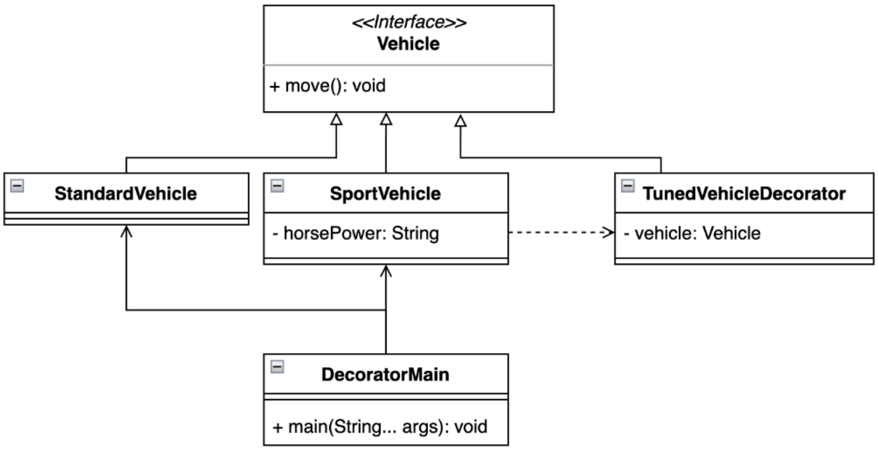

DependencyInjectionMain
+ main(args: String[]): void
FastEngine
- type: String
- started: boolean
EngineServiceProvider
- ENGINES: Map<String, Engine>
+ addEngine(e: Engine): void
+ getEngineByType(type: String): Engine
<<Interface>>
Vehicle
+ move(): void
<<Interface>>
Engine
+ start(): void
+ isStarted(): boolean
+ run(): void
+ type(): String
SportVehicle
- engine: Engine

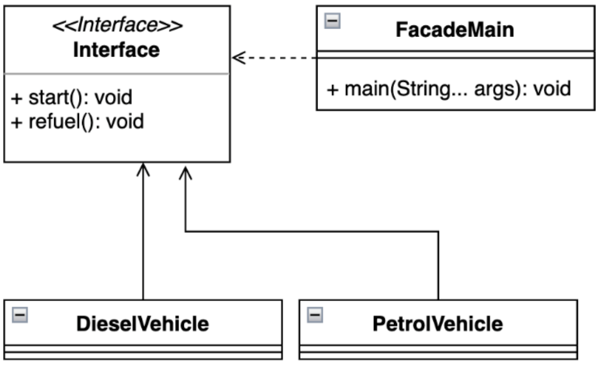
Chapter 4: Applying Structural Design Patterns

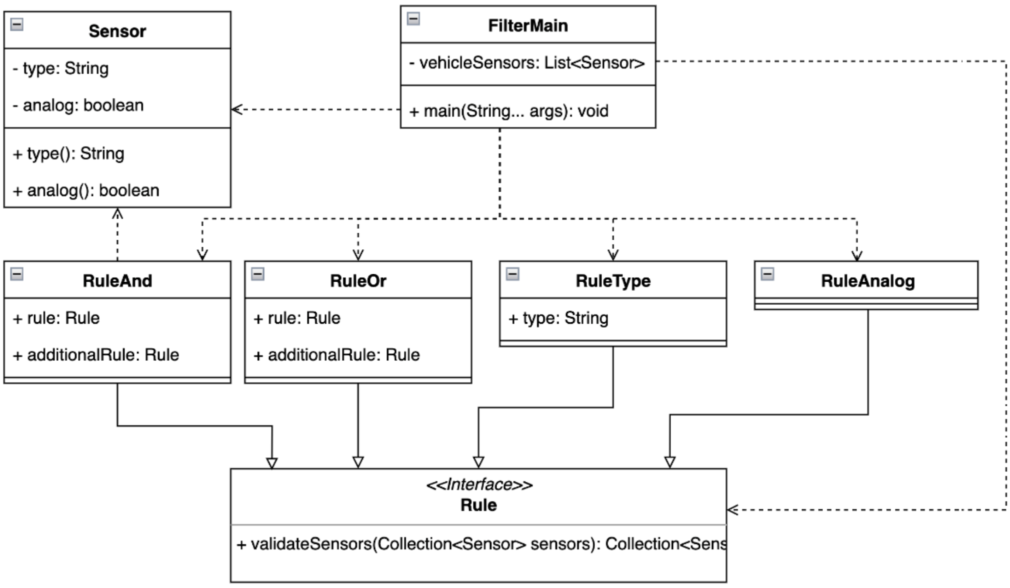


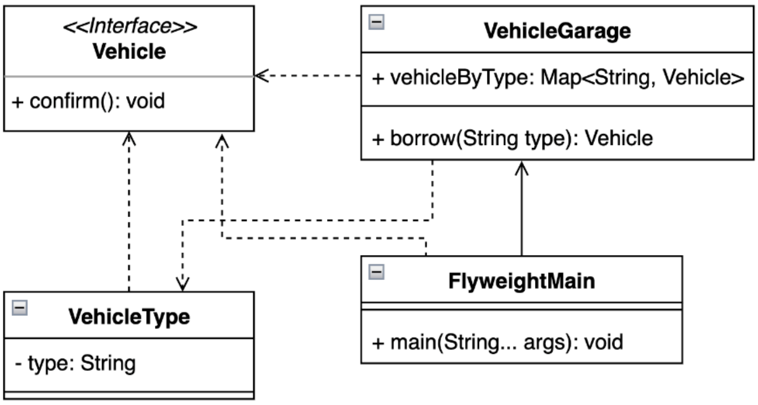


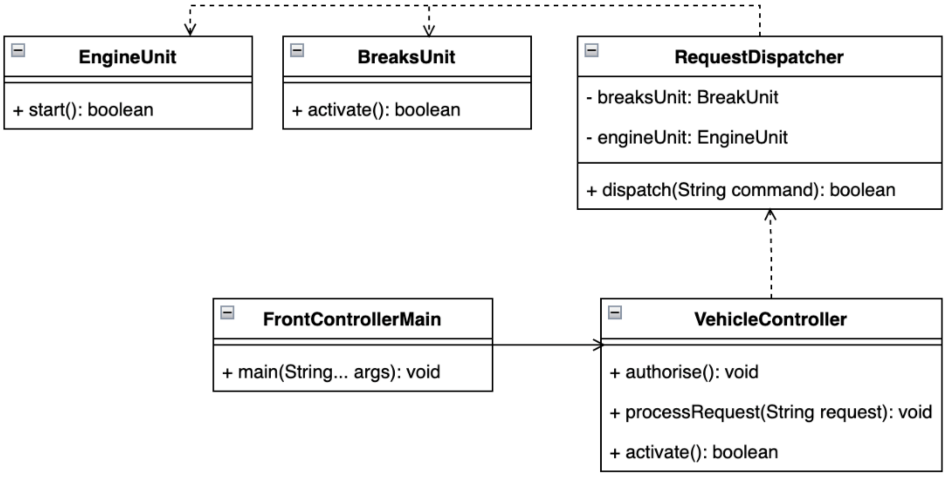

VehiclePart
- type: String
- parts: List<VehiclePart>
+ addPart(VehiclePart p): void
<<Interface>>
VehicleElement
+ addPart(VehicleElement p): void
SportVehicle
- type: String
- parts: List<VehiclePart>
+ addPart(VehiclePart p): void
+ printParts(): void
<<Interface>>
Vehicle
+ printParts(): void
CompositeMain
+ main(String... args): void

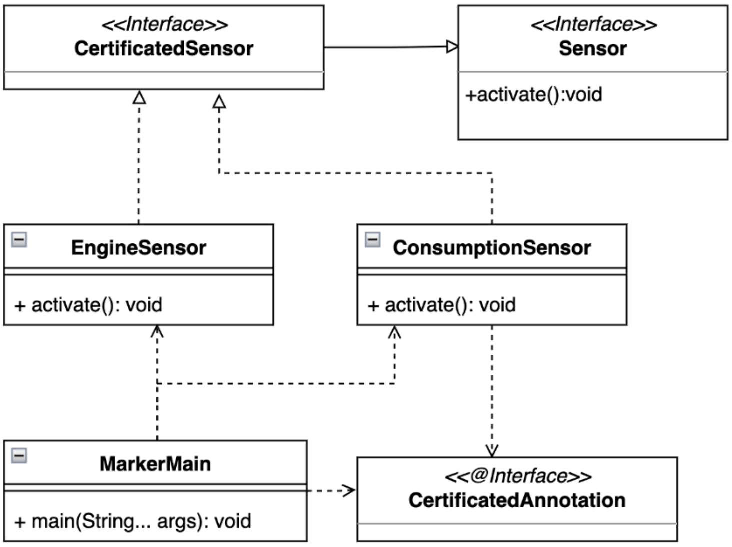

<<Interface>>
Vehicle
+ move(): void
StandardVehicle
SportVehicle
- horsePower: String
TunedVehicleDecorator
- vehicle: Vehicle
DecoratorMain
+ main(String... args): void

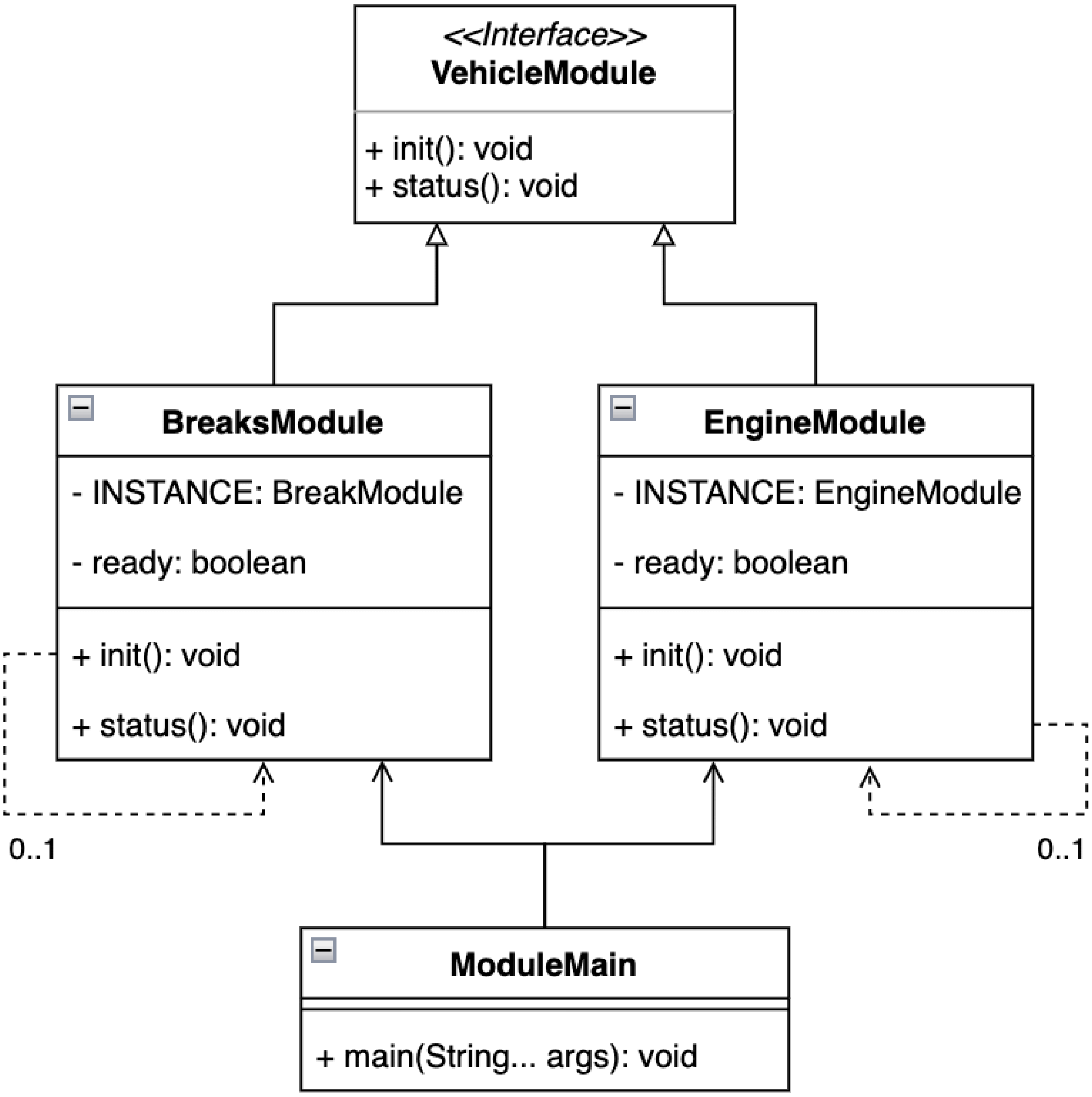

<<Interface>>
Interface
+ start(): void
+ refuel(): void
FacadeMain
+ main(String... args): void
DieselVehicle
PetrolVehicle

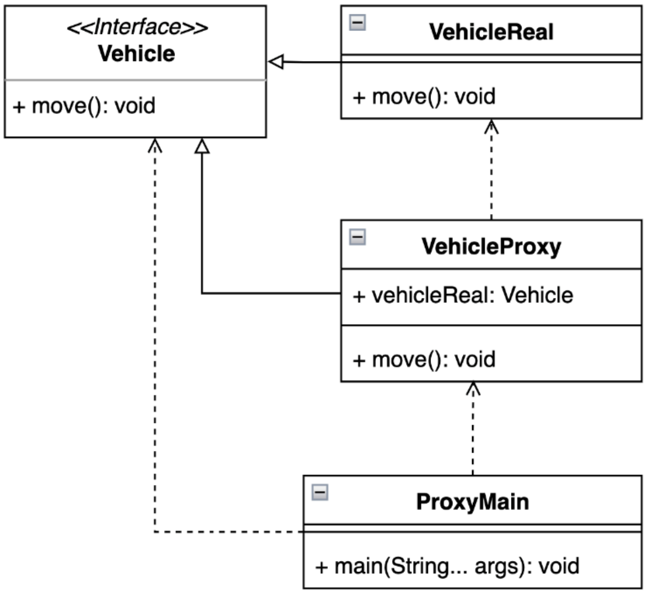

Sensor
- type: String
- analog: boolean
+ type(): String
+ analog(): boolean
FilterMain
- vehicleSensors: List<Sensor>
+ main(String... args): void
RuleAnd
+ rule: Rule
+ additionalRule: Rule
RuleOr
+ rule: Rule
+ additionalRule: Rule
RuleType
+ type: String
RuleAnalog
<<Interface>>
Rule
+ validateSensors(Collection<Sensor> sensors): Collection<Sens

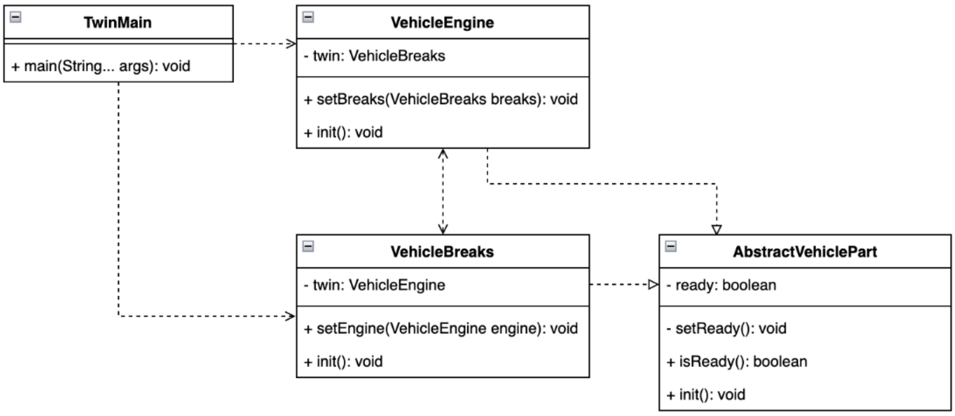

<<Interface>>
Vehicle
+ confirm(): void
VehicleGarage
+ vehicleByType: Map<String, Vehicle>
+ borrow(String type): Vehicle
FlyweightMain
+ main(String... args): void
VehicleType
- type: String


EngineUnit
+ start(): boolean
BreaksUnit
+ activate(): boolean
RequestDispatcher
- breaksUnit: BreakUnit
- engineUnit: EngineUnit
+ dispatch(String command): boolean
FrontControllerMain
+ main(String... args): void
VehicleController
+ authorise(): void
+ processRequest(String request): void
+ activate(): boolean

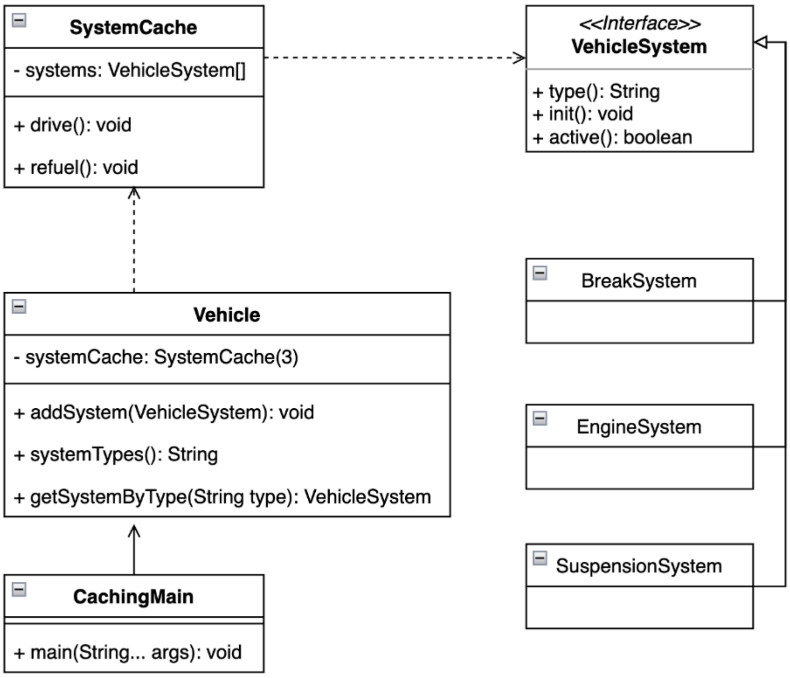

<<Interface>>
CertificatedSensor
<<Interface>>
Sensor
+activate():void
EngineSensor
+ activate(): void
ConsumptionSensor
+ activate(): void
MarkerMain
+ main(String... args): void
<<@Interface>>
CertificatedAnnotation

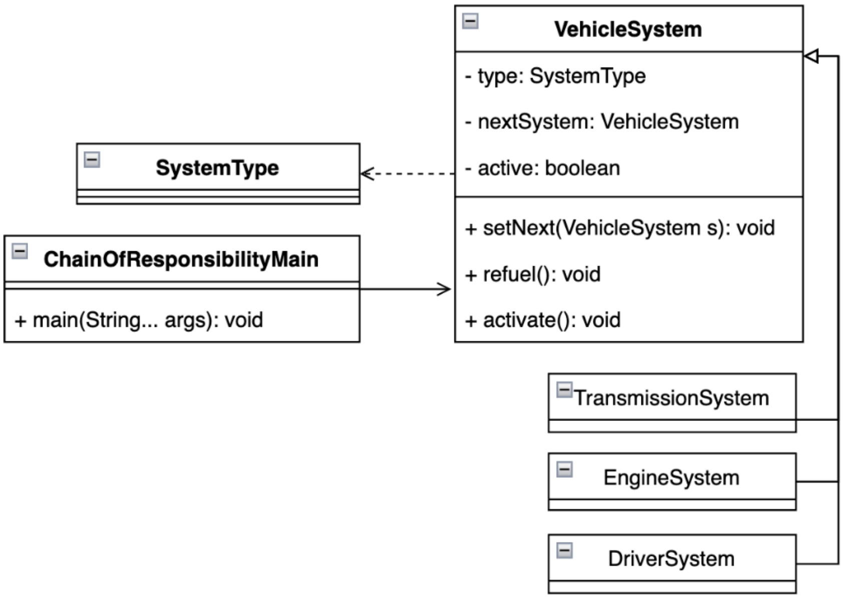

<<Interface>>
VehicleModule
+ init(): void
+ status(): void
BreaksModule
- INSTANCE: BreakModule
- ready: boolean
+ init(): void
+ status(): void
EngineModule
- INSTANCE: EngineModule
- ready: boolean
+ init(): void
+ status(): void
0..1
0..1
ModuleMain
+ main(String... args): void

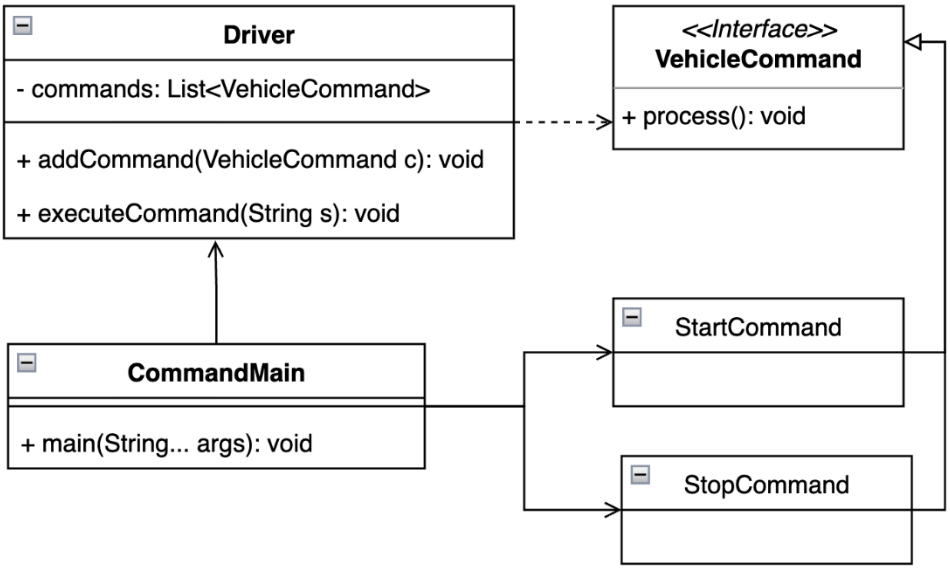

<<Interface>>
Vehicle
+ move(): void
VehicleReal
+ move(): void
VehicleProxy
+ vehicleReal: Vehicle
+ move(): void
ProxyMain
+ main(String... args): void

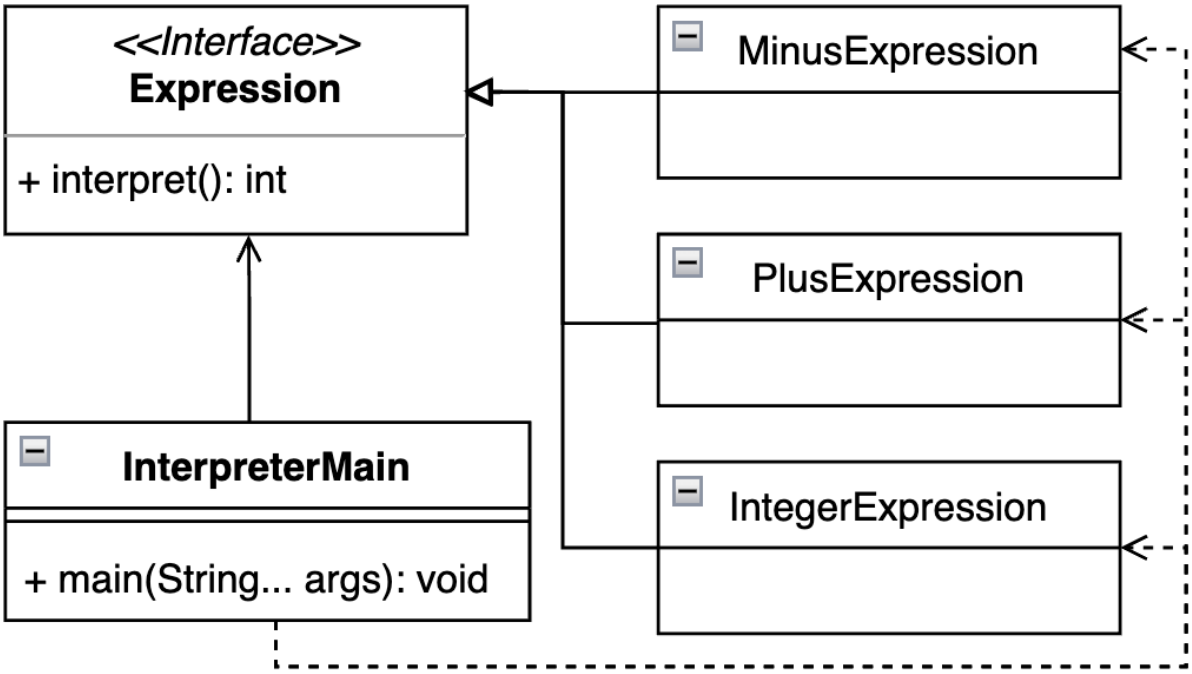

TwinMain
+ main(String... args): void
VehicleEngine
- twin: VehicleBreaks
+ setBreaks(VehicleBreaks breaks): void
+ init(): void
VehicleBreaks
- twin: VehicleEngine
+ setEngine(VehicleEngine engine): void
+ init(): void
AbstractVehiclePart
- ready: boolean
- setReady(): void
+ isReady(): boolean
+ init(): void

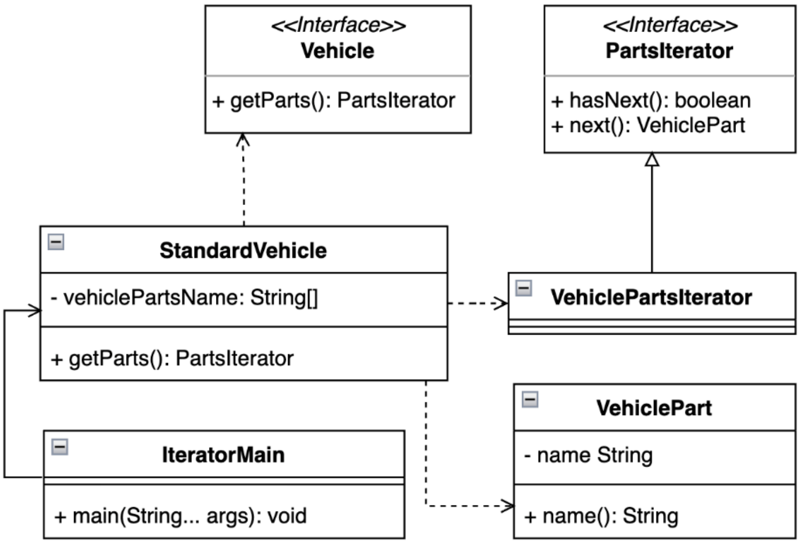
Chapter 5: Behavioral Design Patterns



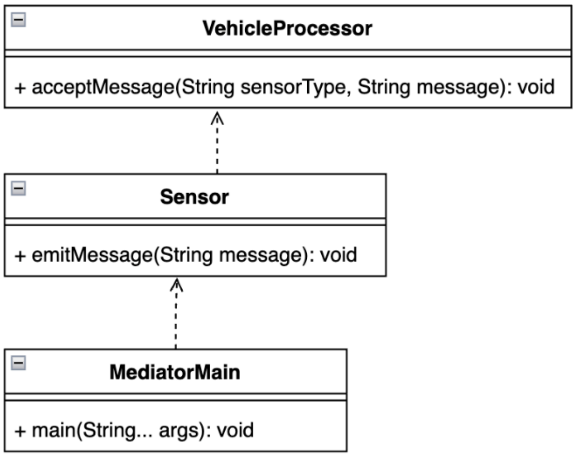



Driver
- commands: List<VehicleCommand>
+ addCommand(VehicleCommand c): void
+ executeCommand(String s): void
<<Interface>>
VehicleCommand
+ process(): void
CommandMain
+ main(String... args): void
StartCommand
StopCommand

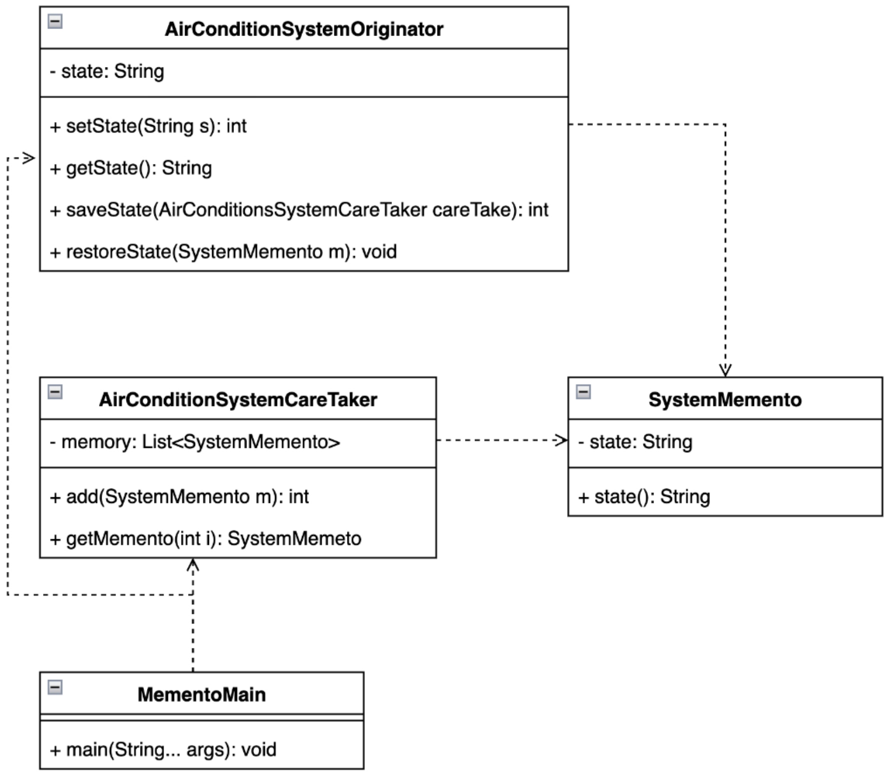

<<Interface>>
Expression
+ interpret(): int
MinusExpression
PlusExpression
IntegerExpression
InterpreterMain
+ main(String... args): void

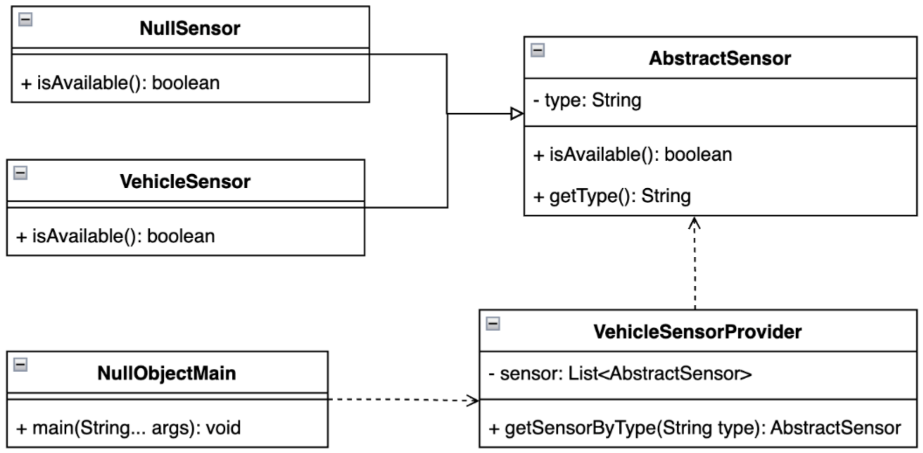


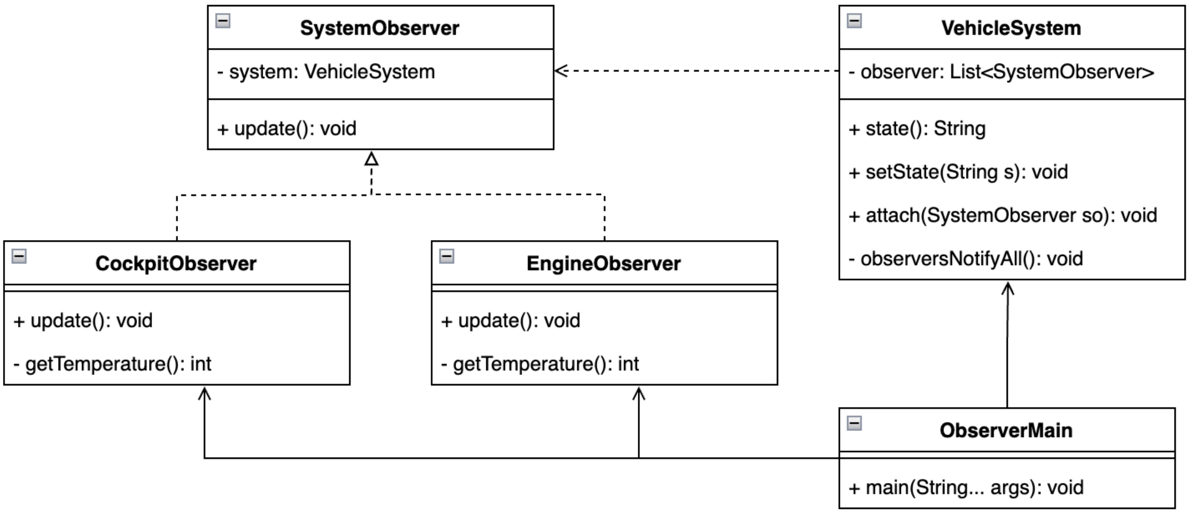
<<Interface>>
Vehicle
+ getParts(): PartsIterator
<<Interface>>
PartsIterator
+ hasNext(): boolean
+ next(): VehiclePart
StandardVehicle
- vehiclePartsName: String[]
+ getParts(): PartsIterator
VehiclePartsIterator
VehiclePart
- name String
+ name(): String
IteratorMain
+ main(String... args): void

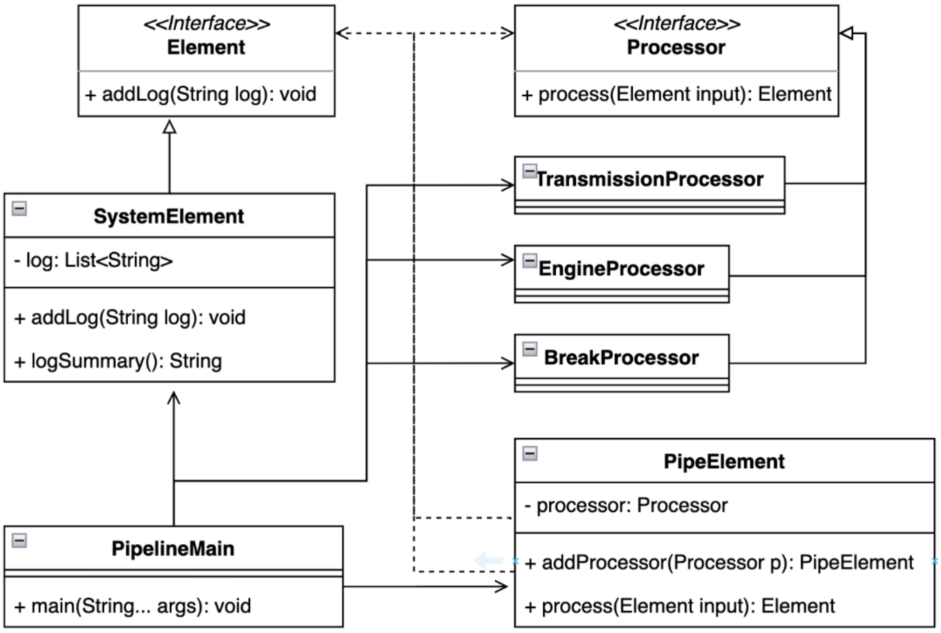


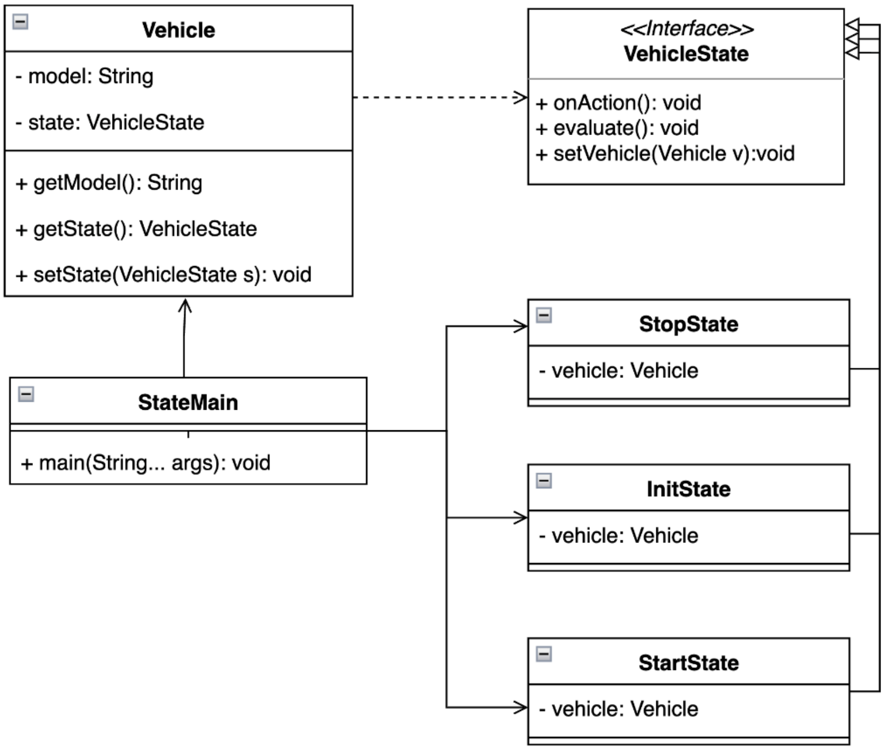
VehicleProcessor
+ acceptMessage(String sensorType, String message): void
Sensor
+ emitMessage(String message): void
MediatorMain
+ main(String... args): void

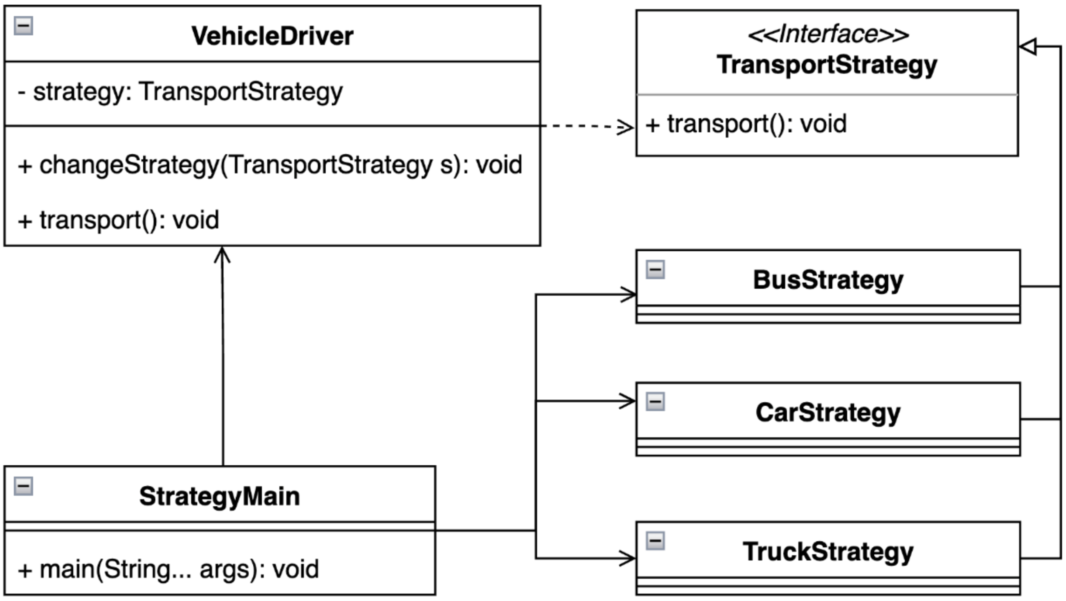

AirConditionSystemOriginator
- state: String
+ setState(String s): int
+ getState(): String
+ saveState(AirConditionsSystemCareTaker careTake): int
+ restoreState(SystemMemento m): void
AirConditionSystemCareTaker
- memory: List<SystemMemento>
+ add(SystemMemento m): int
+ getMemento(int i): SystemMemeto
SystemMemento
- state: String
+ state(): String
MementoMain
+ main(String... args): void

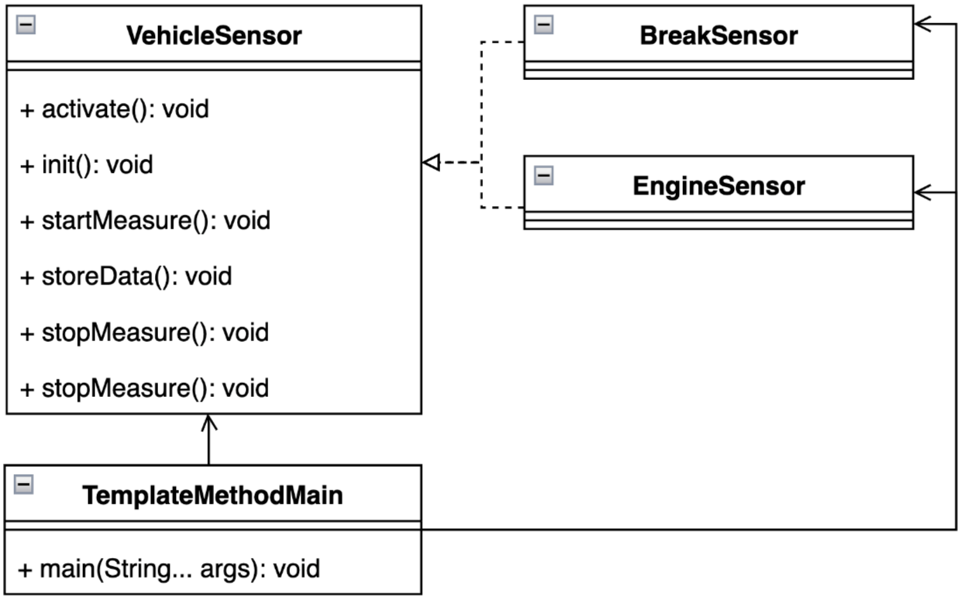

NullSensor
+ isAvailable(): boolean
VehicleSensor
+ isAvailable(): boolean
AbstractSensor
- type: String
+ isAvailable(): boolean
+ getType(): String
NullObjectMain
+ main(String... args): void
VehicleSensorProvider
- sensor: List<AbstractSensor>
+ getSensorByType(String type): AbstractSensor

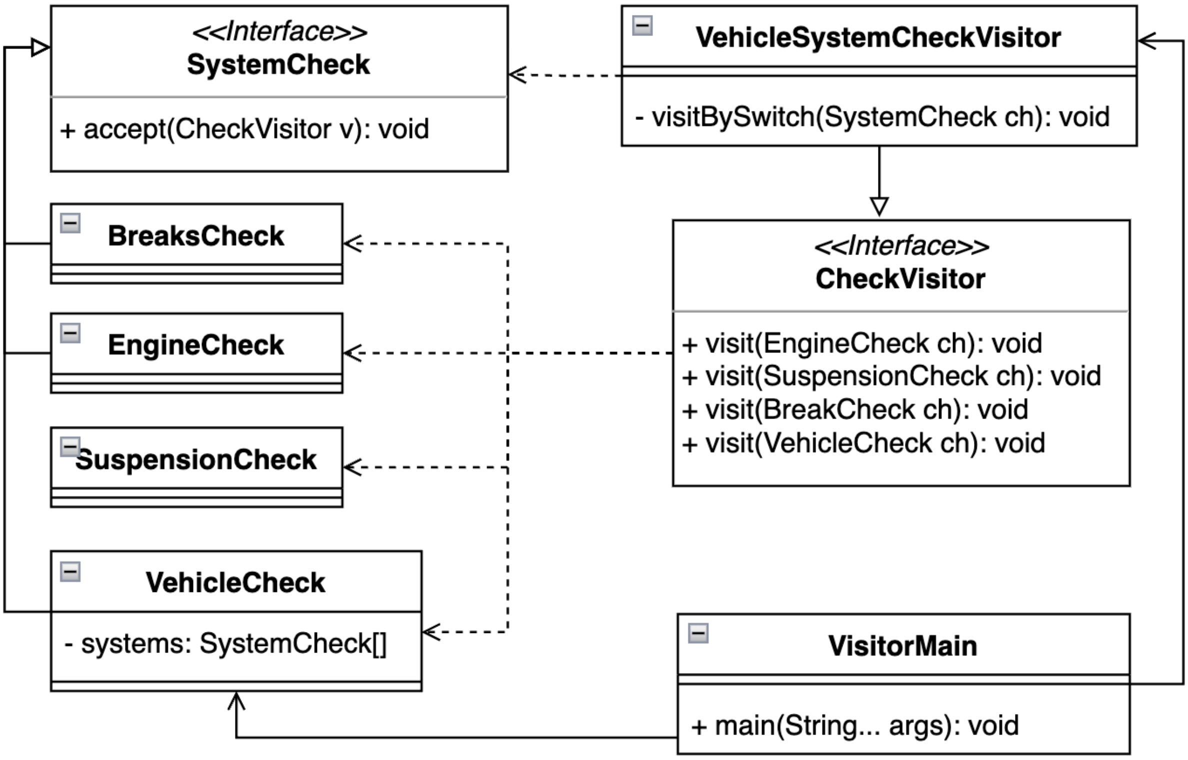

SystemObserver
- system: VehicleSystem
+ update(): void
VehicleSystem
- observer: List<SystemObserver>
+ state(): String
+ setState(String s): void
+ attach(SystemObserver so): void
- observersNotifyAll(): void
CockpitObserver
+ update(): void
- getTemperature(): int
EngineObserver
+ update(): void
- getTemperature(): int
ObserverMain
+ main(String... args): void


<<Interface>>
Element
+ addLog(String log): void
<<Interface>>
Processor
+ process(Element input): Element
TransmissionProcessor
EngineProcessor
BreakProcessor
SystemElement
- log: List<String>
+ addLog(String log): void
+ logSummary(): String
PipeElement
- processor: Processor
+ addProcessor(Processor p): PipeElement
+ process(Element input): Element
PipelineMain
+ main(String... args): void

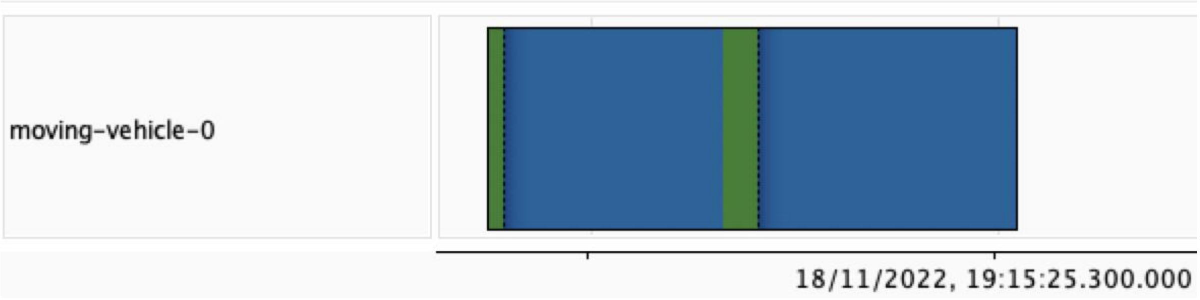

Vehicle
- model: String
- state: VehicleState
+ getModel(): String
+ getState(): VehicleState
+ setState(VehicleState s): void
<<Interface>>
VehicleState
+ onAction(): void
+ evaluate(): void
+ setVehicle(Vehicle v):void
StateMain
+ main(String... args): void
StopState
- vehicle: Vehicle
InitState
- vehicle: Vehicle
StartState
- vehicle: Vehicle

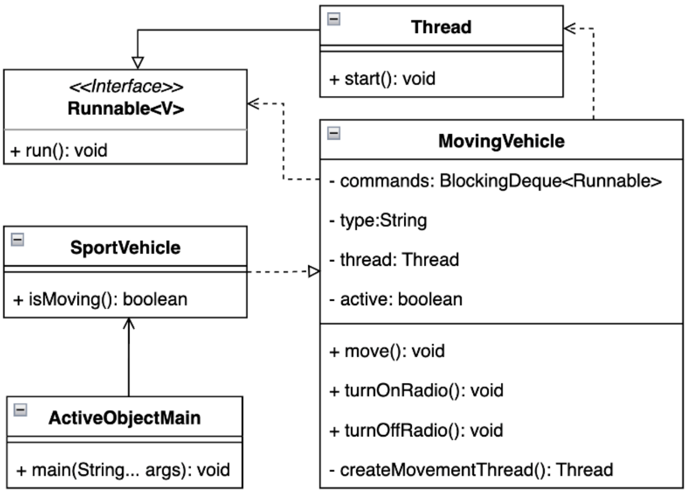

VehicleDriver
- strategy: TransportStrategy
+ changeStrategy(TransportStrategy s): void
+ transport(): void
<<Interface>>
TransportStrategy
+ transport(): void
StrategyMain
+ main(String... args): void
BusStrategy
CarStrategy
TruckStrategy

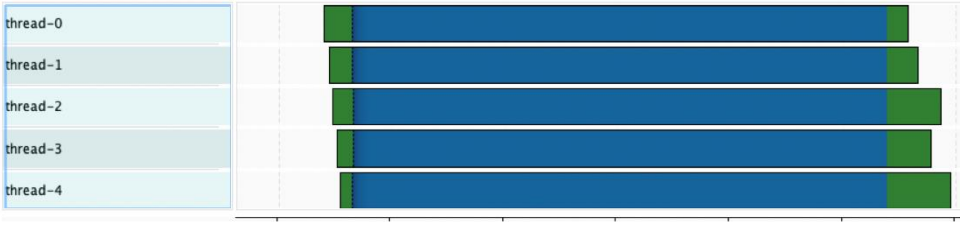

VehicleSensor
+ activate(): void
+ init(): void
+ startMeasure(): void
+ storeData(): void
+ stopMeasure(): void
+ stopMeasure(): void
BreakSensor
EngineSensor
TemplateMethodMain
+ main(String... args): void

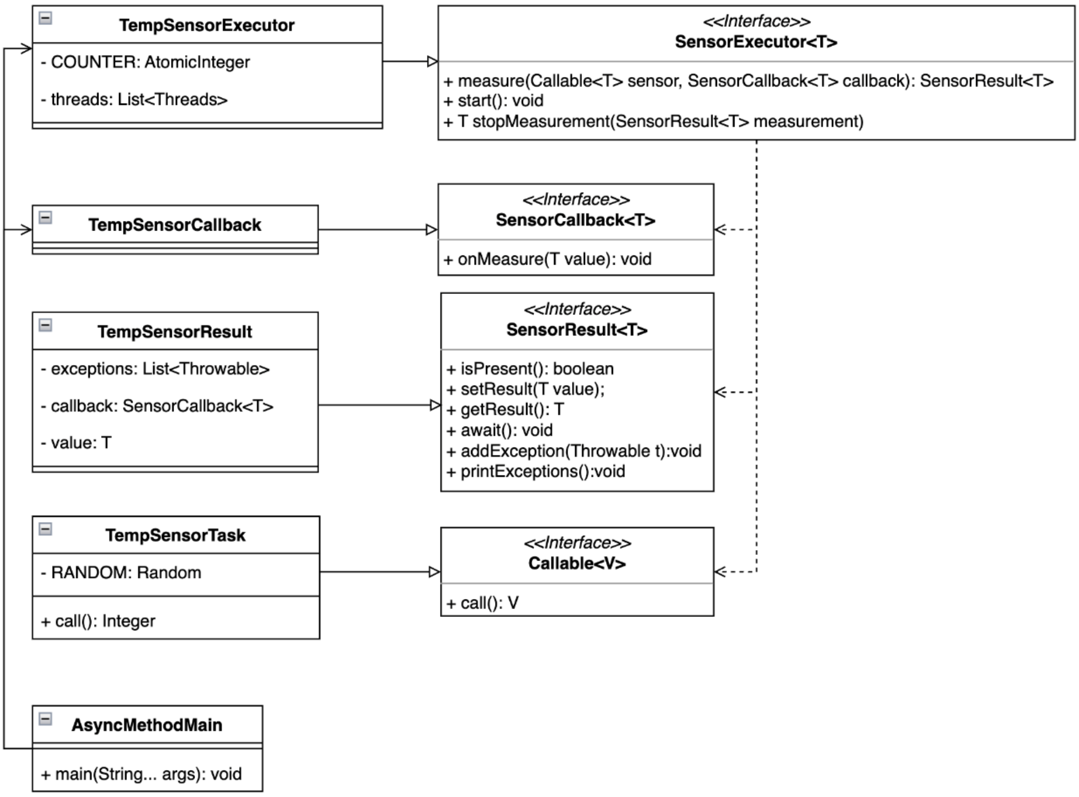

<<Interface>>
SystemCheck
+ accept(CheckVisitor v): void
VehicleSystemCheckVisitor
- visitBySwitch(SystemCheck ch): void
BreaksCheck
EngineCheck
SuspensionCheck
<<Interface>>
CheckVisitor
+ visit(EngineCheck ch): void
+ visit(SuspensionCheck ch): void
+ visit(BreakCheck ch): void
+ visit(VehicleCheck ch): void
VehicleCheck
- systems: SystemCheck[]
VisitorMain
+ main(String... args): void

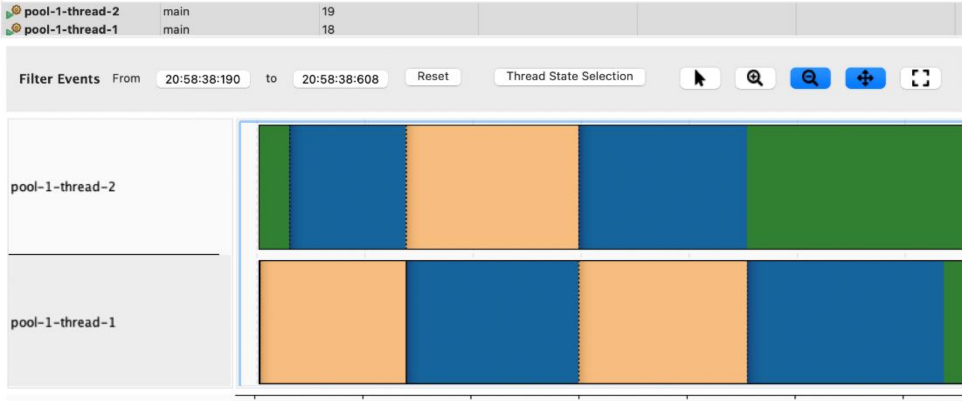
Chapter 6: Concurrency Design Patterns





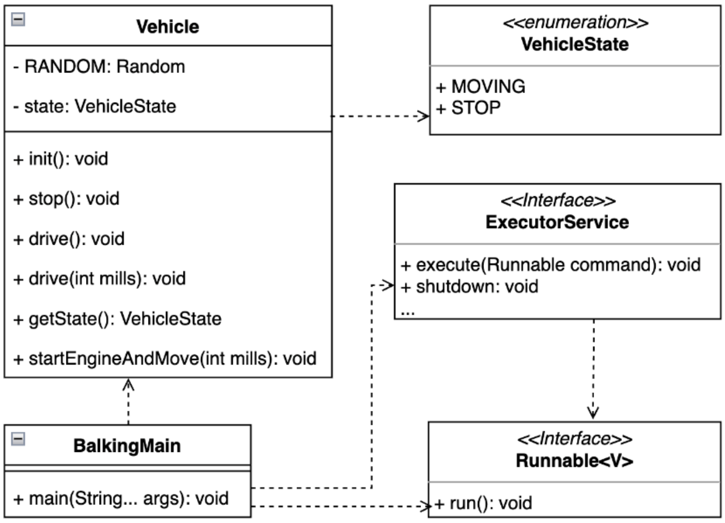


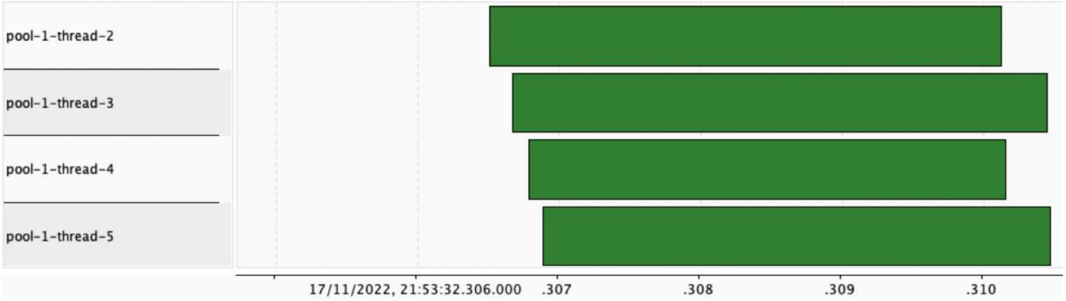


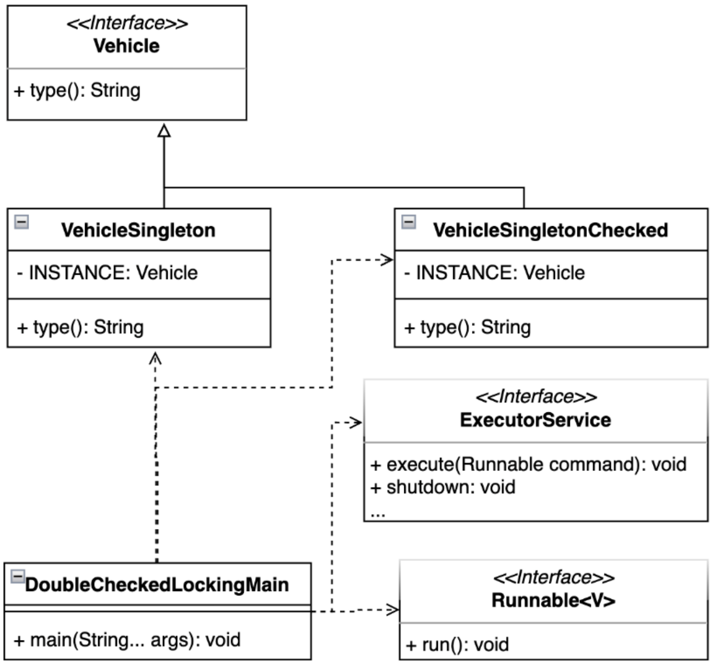


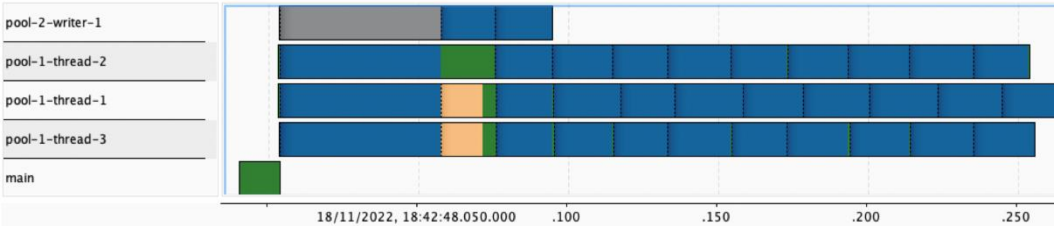
```
pool-1-thread-2    main    19
pool-1-thread-1    main    18
```

Filter Events From 20:58:38:190 to 20:58:38:608

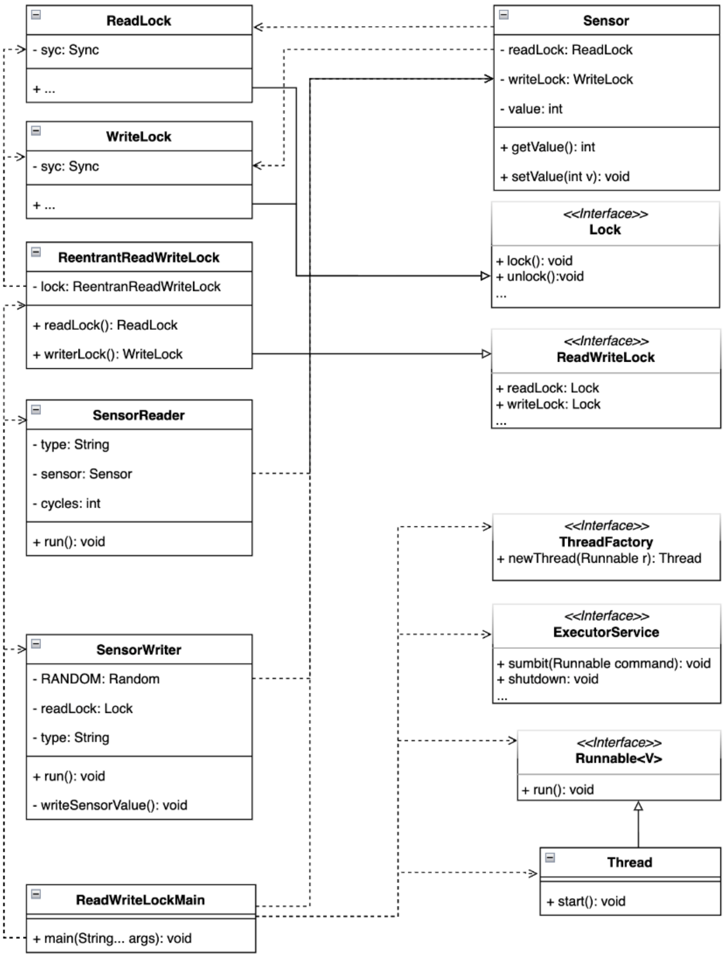

Vehicle
- RANDOM: Random
- state: VehicleState
+ init(): void
+ stop(): void
+ drive(): void
+ drive(int mills): void
+ getState(): VehicleState
+ startEngineAndMove(int mills): void
<<enumeration>>
VehicleState
+ MOVING
+ STOP
<<Interface>>
ExecutorService
+ execute(Runnable command): void
+ shutdown: void
...
BalkingMain
+ main(String... args): void
<<Interface>>
Runnable<V>
+ run(): void


pool-1-thread-2
pool-1-thread-3
pool-1-thread-4
pool-1-thread-5
17/11/2022, 21:53:32.306.000
.307
.308
.309
.310

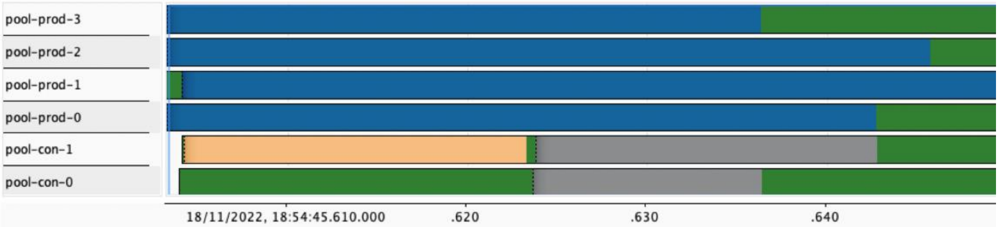

<<Interface>>
Vehicle
+ type(): String
VehicleSingleton
- INSTANCE: Vehicle
+ type(): String
VehicleSingletonChecked
- INSTANCE: Vehicle
+ type(): String
<<Interface>>
ExecutorService
+ execute(Runnable command): void
+ shutdown: void
...
DoubleCheckedLockingMain
+ main(String... args): void
<<Interface>>
Runnable<V>
+ run(): void



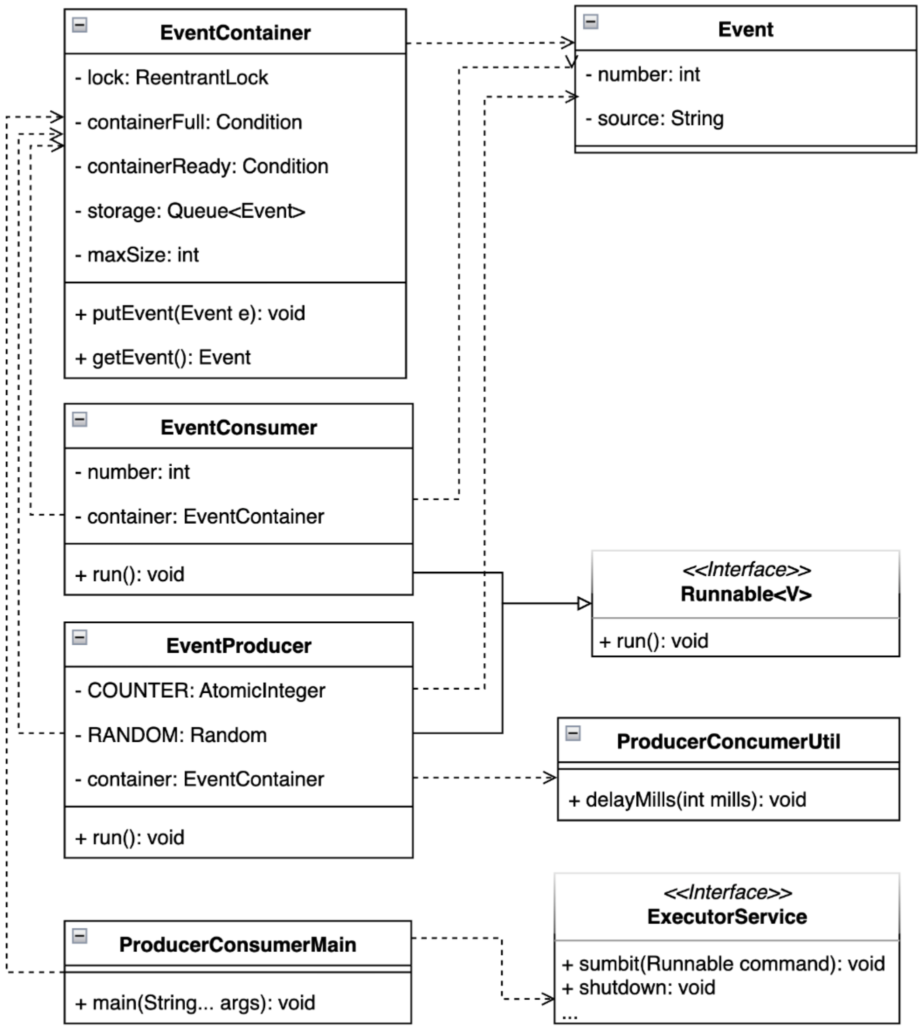
pool-2-writer-1
pool-1-thread-2
pool-1-thread-1
pool-1-thread-3
main
18/11/2022, 18:42:48.050.000
.100
.150
.200
.250

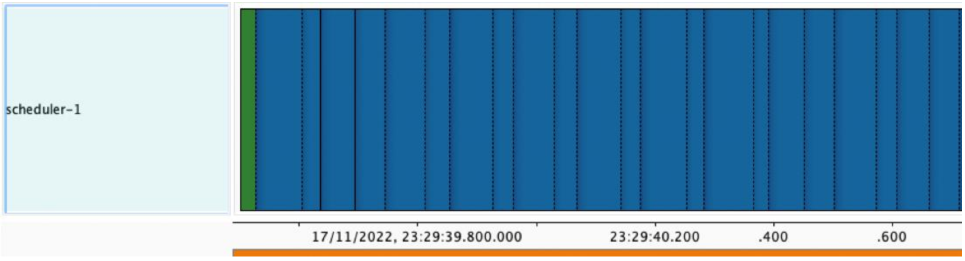


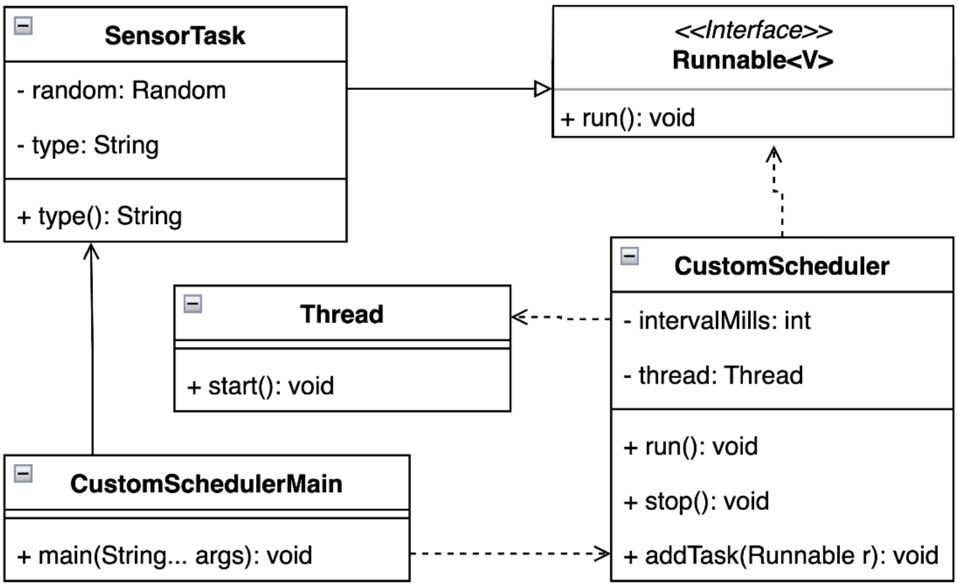
ReadLock
- syc: Sync
+ ...
WriteLock
- syc: Sync
+ ...
ReentrantReadWriteLock
- lock: ReentranReadWriteLock
+ readLock(): ReadLock
+ writerLock(): WriteLock
SensorReader
- type: String
- sensor: Sensor
- cycles: int
+ run(): void
SensorWriter
- RANDOM: Random
- readLock: Lock
- type: String
+ run(): void
- writeSensorValue(): void
ReadWriteLockMain
+ main(String... args): void
Sensor
- readLock: ReadLock
- writeLock: WriteLock
- value: int
+ getValue(): int
+ setValue(int v): void
<<Interface>>
Lock
+ lock(): void
+ unlock():void
...
<<Interface>>
ReadWriteLock
+ readLock: Lock
+ writeLock: Lock
...
<<Interface>>
ThreadFactory
+ newThread(Runnable r): Thread
<<Interface>>
ExecutorService
+ sumbit(Runnable command): void
+ shutdown: void
...
<<Interface>>
Runnable<V>
+ run(): void
Thread
+ start(): void

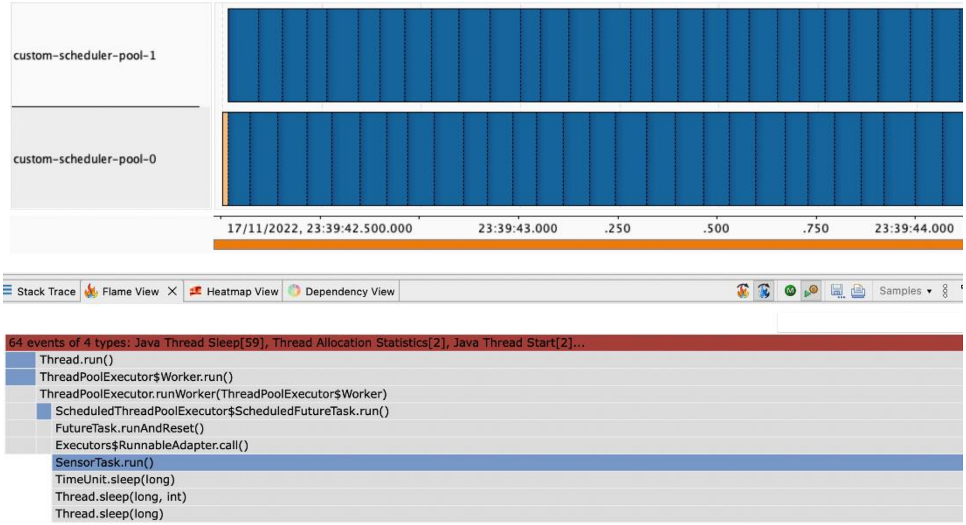


pool-prod-3
pool-prod-2
pool-prod-1
pool-prod-0
pool-con-1
pool-con-0
18/11/2022, 18:54:45.610.000
.620
.630
.640

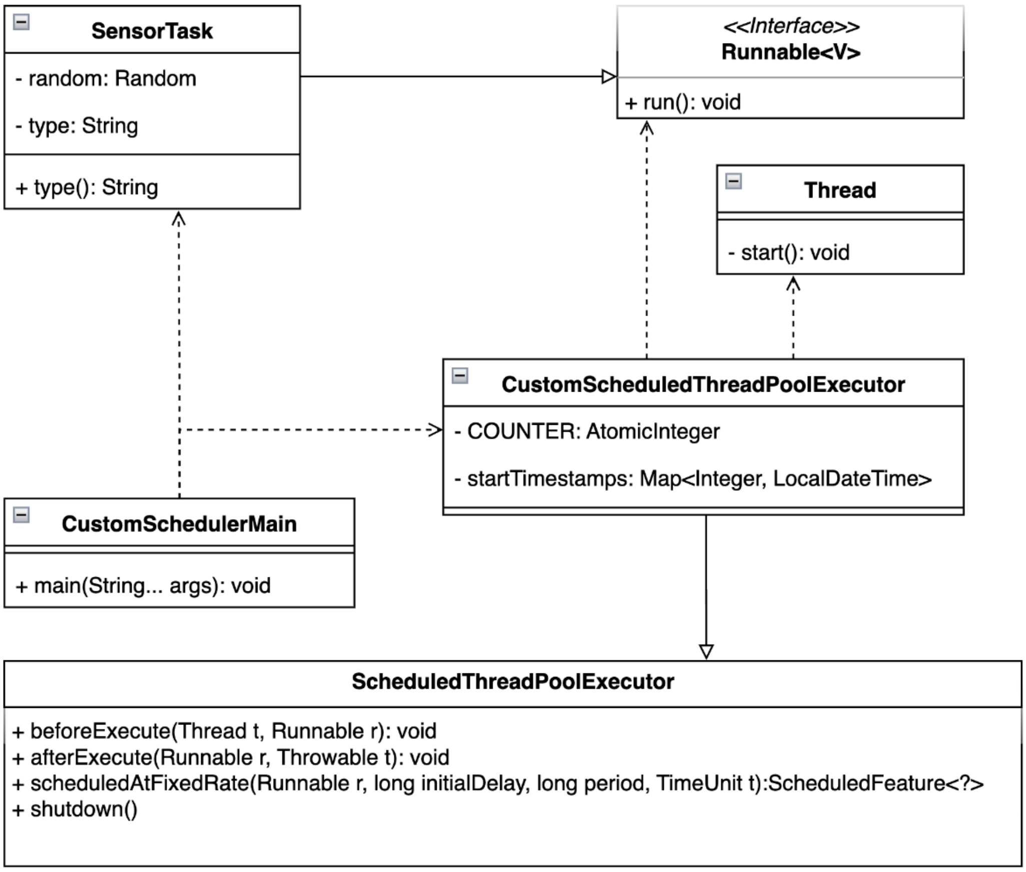

EventContainer
- lock: ReentrantLock
- containerFull: Condition
- containerReady: Condition
- storage: Queue<Event>
- maxSize: int
+ putEvent(Event e): void
+ getEvent(): Event
Event
- number: int
- source: String
EventConsumer
- number: int
- container: EventContainer
+ run(): void
<<Interface>>
Runnable<V>
+ run(): void
EventProducer
- COUNTER: AtomicInteger
- RANDOM: Random
- container: EventContainer
+ run(): void
ProducerConcumerUtil
+ delayMills(int mills): void
<<Interface>>
ExecutorService
+ sumbit(Runnable command): void
+ shutdown: void
...
ProducerConsumerMain
+ main(String... args): void

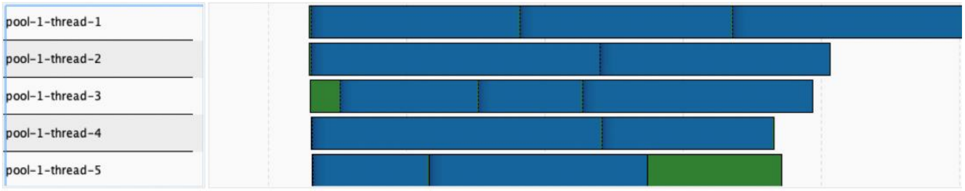

scheduler-1
17/11/2022, 23:29:39.800.000
23:29:40.200
.400
.600

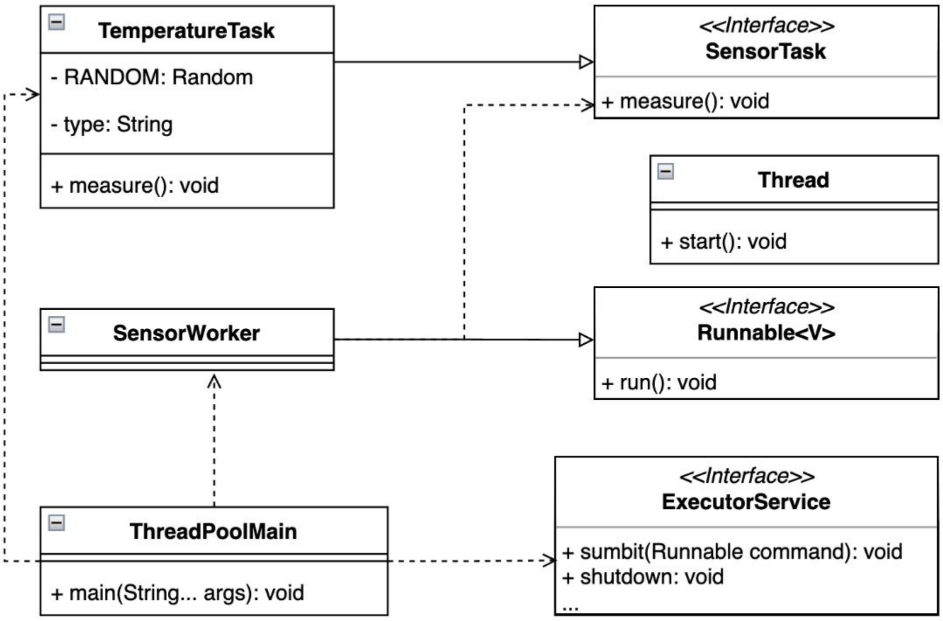




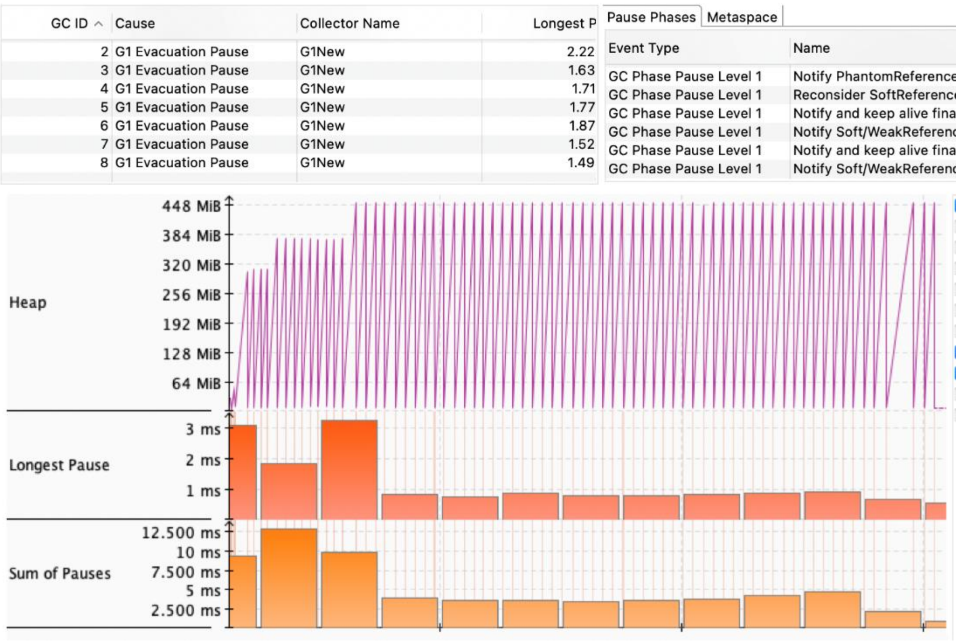
```
64 events of 4 types: Java Thread Sleep[59], Thread Allocation Statistics[2], Java Thread Start[2]...
Thread.run()
ThreadPoolExecutor$Worker.run()
ThreadPoolExecutor.runWorker(ThreadPoolExecutor$Worker)
ScheduledThreadPoolExecutor$ScheduledFutureTask.run()
FutureTask.runAndReset()
Executors$RunnableAdapter.call()
SensorTask.run()
TimeUnit.sleep(long)
Thread.sleep(long, int)
Thread.sleep(long)
```


SensorTask
- random: Random
- type: String
+ type(): String
<<Interface>>
Runnable<V>
+ run(): void
Thread
- start(): void
CustomScheduledThreadPoolExecutor
- COUNTER: AtomicInteger
- startTimestamps: Map<Integer, LocalDateTime>
CustomSchedulerMain
+ main(String... args): void
ScheduledThreadPoolExecutor
+ beforeExecute(Thread t, Runnable r): void
+ afterExecute(Runnable r, Throwable t): void
+ scheduledAtFixedRate(Runnable r, long initialDelay, long period, TimeUnit t):ScheduledFeature<?>
+ shutdown()


pool-1-thread-1
pool-1-thread-2
pool-1-thread-3
pool-1-thread-4
pool-1-thread-5


TemperatureTask
- RANDOM: Random
- type: String
+ measure(): void
<<Interface>>
SensorTask
+ measure(): void
Thread
+ start(): void
SensorWorker
<<Interface>>
Runnable<V>
+ run(): void
ThreadPoolMain
+ main(String... args): void
<<Interface>>
ExecutorService
+ sumbit(Runnable command): void
+ shutdown: void
...

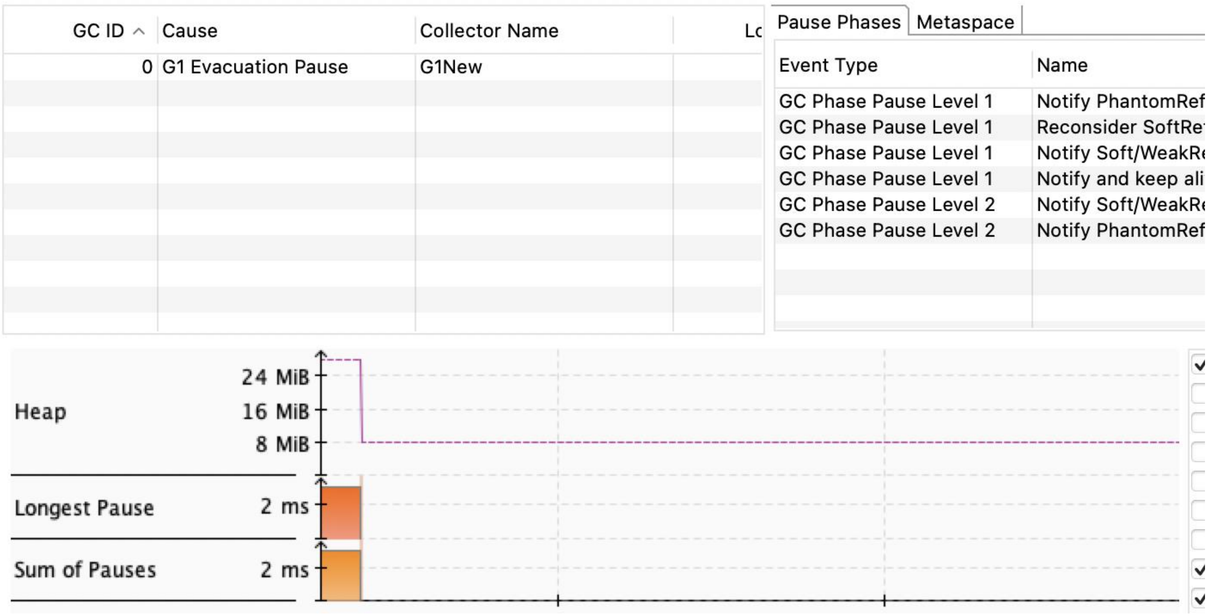
Chapter 7: Understanding Common Anti-Patterns



GC ID	Cause	Collector Name	Longest P
2	G1 Evacuation Pause	G1New	2.22
3	G1 Evacuation Pause	G1New	1.63
4	G1 Evacuation Pause	G1New	1.71
5	G1 Evacuation Pause	G1New	1.77
6	G1 Evacuation Pause	G1New	1.87
7	G1 Evacuation Pause	G1New	1.52
8	G1 Evacuation Pause	G1New	1.49

Pause Phases | Metaspace

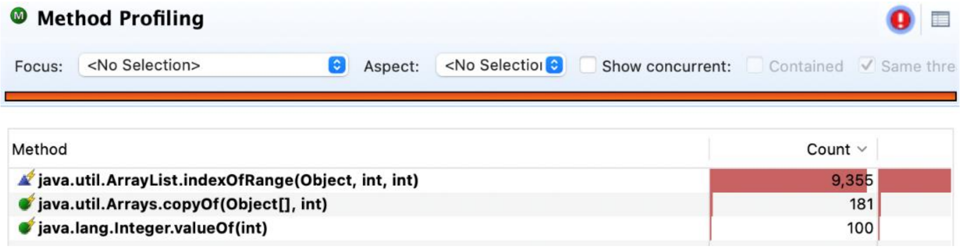
Event Type	Name
GC Phase Pause Level 1	Notify PhantomReference
GC Phase Pause Level 1	Reconsider SoftReferenc
GC Phase Pause Level 1	Notify and keep alive fina
GC Phase Pause Level 1	Notify Soft/WeakReferenc
GC Phase Pause Level 1	Notify and keep alive fina
GC Phase Pause Level 1	Notify Soft/WeakReferenc



GC ID	Cause	Collector Name	Lc
0	G1 Evacuation Pause	G1New	

Pause Phases | Metaspace

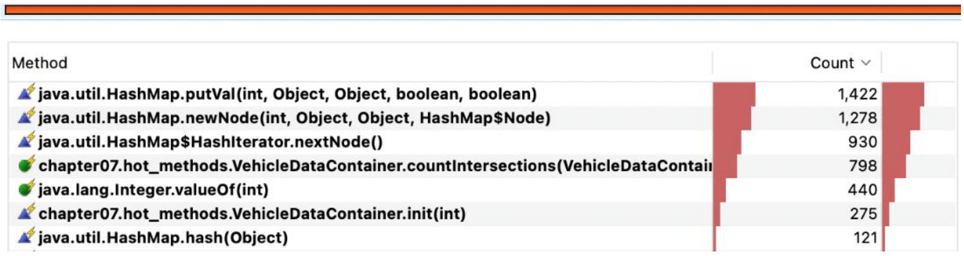
Event Type	Name
GC Phase Pause Level 1	Notify PhantomRef
GC Phase Pause Level 1	Reconsider SoftRe
GC Phase Pause Level 1	Notify Soft/WeakR
GC Phase Pause Level 1	Notify and keep ali
GC Phase Pause Level 2	Notify Soft/WeakR
GC Phase Pause Level 2	Notify PhantomRef



Method Profiling

Focus: <No Selection> Aspect: <No Selectio ☐ Show concurrent: ☐ Contained ☑ Same thre

Method	Count
java.util.ArrayList.indexOfRange(Object, int, int)	9,355
java.util.Arrays.copyOf(Object[], int)	181
java.lang.Integer.valueOf(int)	100



Method	Count
java.util.HashMap.putVal(int, Object, Object, boolean, boolean)	1,422
java.util.HashMap.newNode(int, Object, Object, HashMap$Node)	1,278
java.util.HashMap$HashIterator.nextNode()	930
chapter07.hot_methods.VehicleDataContainer.countIntersections(VehicleDataContai	798
java.lang.Integer.valueOf(int)	440
chapter07.hot_methods.VehicleDataContainer.init(int)	275
java.util.HashMap.hash(Object)	121