AN INTRODUCTION TO AKKA PERSISTENCE

Bartosz Mikulski

THE SPEAKER

@mikulskibartosz

- Java / Scala / sometimes JS ;)
- programmer for 5 years
- co-founder of Poznan Scala User Group
  meetup.com/poznanscala

EVENT SOURCING

1. An actor receives a command
2. The actor checks whether the command can be applied to the current state
3. The actor generates and stores an event

An event describes the effect of a command

EVENT SOURCING

- commands
- events
- Command Query Responsibility Segregation

EVENT SOURCING - EXTERNAL UPDATES

- if the application replays events, external system processes duplicate messages
- communicate with external system using a Gateway
- disable the Gateway during recovery

EVENT SOURCING - EXTERNAL QUERIES

- returned value influences event processing
- the value may be different during recovery
- use a Gateway to query external systems
- the Gateway may store value returned from the system

PERSISTENT ACTOR

- val receiveRecover: Receive = ???
- val receiveCommand: Receive = ???

RECEIVECOMMAND

```
case object Snapshot
case class Increase(n: Int)
case class Increased(n: Int)

var state = 0

val receiveCommand: Receive = {
        case Increase(n) =>
                persist(Increased(n)){ event =>
                        state = state + n
                }
        case Snapshot => saveSnapshot(state)
}
```

RECEIVERECOVER

```
val receiveRecover: Receive = {
    case Increased(n) => state = state + n
    case SnapshotOffer(_, snapshot: Int) => state = snapshot
}
```

PERSIST

def persist[A](event: A, handler: Procedure[A]): Unit

- persists event, then runs the handler
- handler can mutate state
- it is safe to call sender()
- message stashing

EVENT STORAGE IDENTIFIER

- mandatory
- identifies collection used to store events
- override def persistenceId = "never, ever change this value"

AT LEAST ONCE DELIVERY

- trait AtLeastOnceDelivery extends PersistentActor with AtLeastOnceDeliveryLike
- def deliver(destination: ActorPath)(deliveryIdToMessage: Long ⇒ Any): Unit
- def confirmDelivery(deliveryId: Long): Boolean
- def redeliverInterval: FiniteDuration
- akka.persistence.at-least-once-delivery.redeliver-interval

AT LEAST ONCE DELIVERY - DELIVERYID

- deliveryId - strictly monotonically increasing sequence number without gaps
- It is not possible to use custom deliveryId
- The trait has a state consisting of unconfirmed messages and a sequence number

AT LEAST ONCE DELIVERY - RECOVERY

- You must store events corresponding to deliver and confirmDelivery calls
- Calls to deliver will not send out messages
- Messages will be sent later if no matching confirmDelivery calls have been performed

SHUTDOWN

- do not use PoisonPill!
- actor would stop before processing stashed messages
- explicit shutdown message
  - case object Stop
  - case object Shutdown
  - case object GoHomeAkkaYouAreDrunk

RELAXED CONSISTENCY REQUIREMENTS

- "(...) process commands as fast as you can, assuming that the Event will eventually be persisted and handled properly in the background, retroactively reacting to persistence failures if needed"
- to implement the "command sourcing" pattern (store commands instead of events)

```
def persistAsync[A](event: A)(handler: Procedure[A]): Unit
```

PERSISTENT QUERY

- receives updates from an actor
- has own storage - identified by the "viewId" parameter

SNAPSHOT

- saveSnapshot
- SaveSnapshotSuccess / SaveSnapshotFailure
- final case class SnapshotMetadata(persistenceId: String, sequenceNr: Long, timestamp: Long = 0L)
- SnapshotOffer(metadata, offeredSnapshot)

SNAPSHOT - RECOVERY

```
override def recovery = Recovery(fromSnapshot = SnapshotSelectionCriteria(
  maxSequenceNr = 457L,
  maxTimestamp = System.currentTimeMillis))
```

- SnapshotSelectionCriteria.Latest - default
- SnapshotSelectionCriteria.None - disable snapshot-based recovery
- if no snapshot matched => replay all messages

RECOVERY

- automatic
- on start and restart
- recovery completed message
- stashes incoming messages

EVENT ADAPTERS

- schema evolution (version migration)
- separating domain and data model
- event adapter bindings (conf file)

TESTING

- akka.persistence.journal.leveldb.native = off, or
- akka.persistence.journal.leveldb-shared.store.native = off
- Not possible to use TestActorRef
- Use TestKit instead

BOOKS AND LINKS

- http://doc.akka.io/docs/akka/snapshot/scala/persistence.htm
- http://martinfowler.com/eaaDev/EventSourcing.html
- Vaughn Vernon - Implementing Domain-Driven Design

WORKSHOP TIME!