Applying Business Logic

Handlers apply business logic at specific points in the pipeline. Handlers are decorated with @AviatorHandler, indicating the transaction type and pipeline stages they are invoked for.

```
@AviatorHandler(namespace=ZooDemoTransactionTypes.NAMESPACE,
                transactionType=ZooDemoTransactionTypes.ADD_ANIMAL,
                events={PlatformEvents.executePreConsensus, PlatformEvents.executeConsensus},
                payloadClass=Animal.class)
public void addAnimal(AviatorMessage<Animal> message, SocketDemoState state) {
    //todo:  improve this so that we're testing if an animal of the same name exists, and failing if so
    Animal animal = message.payload;
    switch (animal.getSpecies()) {
        case "lion":
            state.addLion(animal.getName());
            break;
        case "tiger":
            state.addTiger(animal.getName());
            break;
        case "bear":
            state.addBear(animal.getName());
            break;
    }
}
```

Reporting Results

Subscribers report information about the status of a transaction. Subscribers are decorated with the @AviatorSubscriber annotation, which describes the transaction type and pipeline stages they are invoked for.

```
@AviatorSubscriber(namespace=ZooDemoTransactionTypes.NAMESPACE,
                   transactionType=ZooDemoTransactionTypes.ADD_ANIMAL,
                   events= {
                           ReportingEvents.submitted,
                           ReportingEvents.preConsensusResult,
                           ReportingEvents.consensusResult,
                           ReportingEvents.transactionComplete })
public void addAnimalTransactionProgress(AviatorNotification<?> notification) {
    String myName = PlatformLocator.getState().getMyName();
    System.out.println("Sending notification from " + myName);
    this.sendNotification(notification);
}
```