Lens example 2/3

- We can also “`map()`” the lens into domain types:

```
data class BTC(val value: Int) {
    operator fun plus(that: BTC) = BTC(value + that.value)
    fun toString(): value.toString()
}

val btcPath = Path.int().map(::BTC).of("btc")

val miner: HttpHandler = CatchLensFailure.then(
    routes(
        “/mine/{btc}” bind POST to { r: Request ->
            val newTotal: BTC = btcPath.extract(r) + BTC(1)
            val message = “You have $newTotal BTC”
            val json = """{ "value": "$message"}"""
            Response(OK).body(json)
        }
    ))
```

Lens example 3/3

- Going further, use auto JSON marshalling in Jackson:

```
data class BTC(val value: Int) {
    operator fun plus(that: BTC) = BTC(value + that.value)
    fun toString(): value.toString()
}

class MinedCoin(btc: Btc) {
    val value = “You have ${btc + BTC(1)} BTC”
}

val btcPath = Path.int().map(::BTC).of("btc")
val jsonBody = Body.auto<MinedCoin>().toLens()

val miner: HttpHandler = CatchLensFailure.then(
    routes(
        “/mine/{btc}” bind GET to { r: Request ->
            val mined = MinedCoin(btcPath.extract(r))
            jsonBody.inject(mined, Response(OK))
        }
    ))
```