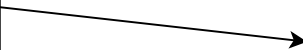## Ownable.sol

origOwner: address

TransferOwnership: event

---

owner()

onlyOwner()

isOwner()

renounceOwnership()

transferOwnership()

_transferOwnership()

SupplyChain.sol

owner: address

upc: unit

sku: unit

items: mapping

itemsHistory: mapping

State: enum

Item: struct

items: mapping

Harvested: event

Processed: event

Packed: event

ForSale: event

Sold: event

Shipped: event

Received: event

Purchased: event

---

onlyOwner()

verifyCaller(_address)

paidEnough(_price)

checkValue(_price)

harvested(_upc)

processed(_upc)

packed(_upc)

forSale(_upc)

sold(_upc)

shipped(_upc)

received(_upc)

purchased(_upc)

kill()

harvestItem()

processItem()

packItem()

sellItem()

buyItem()

shipItem()

receiveItem()

purchaseItem()

fetchItemBufferOne()

fetchItemBufferTwo()

FarmerRole.sol

farmers: struct

FarmerAdded: event

FarmerRemoved: event

---

onlyFarmer()

isFarmer(account): bool

addFarmer(account)

renounceFarmer()

_addFarmer(account)

_removeFarmer(account)

DistributorRole.sol

distributors: struct

DistributorAdded: event

DistributorRemoved: event

---

onlyDistributor()

isDistributor(account): bool

addDistributor(account)

renounceDistributor()

_addDistributor(account)

_removeDistributor(account)

RetailerRole.sol

retailers: struct

RetailerAdded: event

RetailerRemoved: event

---

onlyRetailer()

isRetailer(account): bool

addRetailer(account)

renounceRetailer()

_addRetailer(account)

_removeRetailer(account)

ConsumerRole.sol

consumers: struct

ConsumerAdded: event

ConsumerRemoved: event

---

onlyConsumer()

isConsumer(account): bool

addConsumer(account)

renounceConsumer()

_addConsumer(account)

_removeConsumer(account)

Roles.sol

bearer: mapping

---

add()

remove()

has()