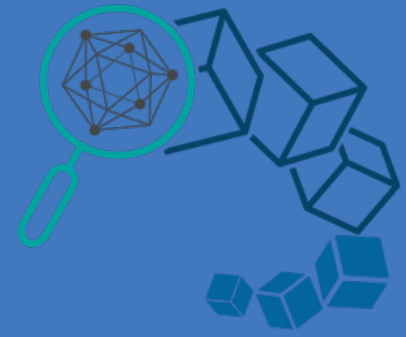




HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS



Hyperledger Fabric Application Development

Baohua Yang
April 7, 2017

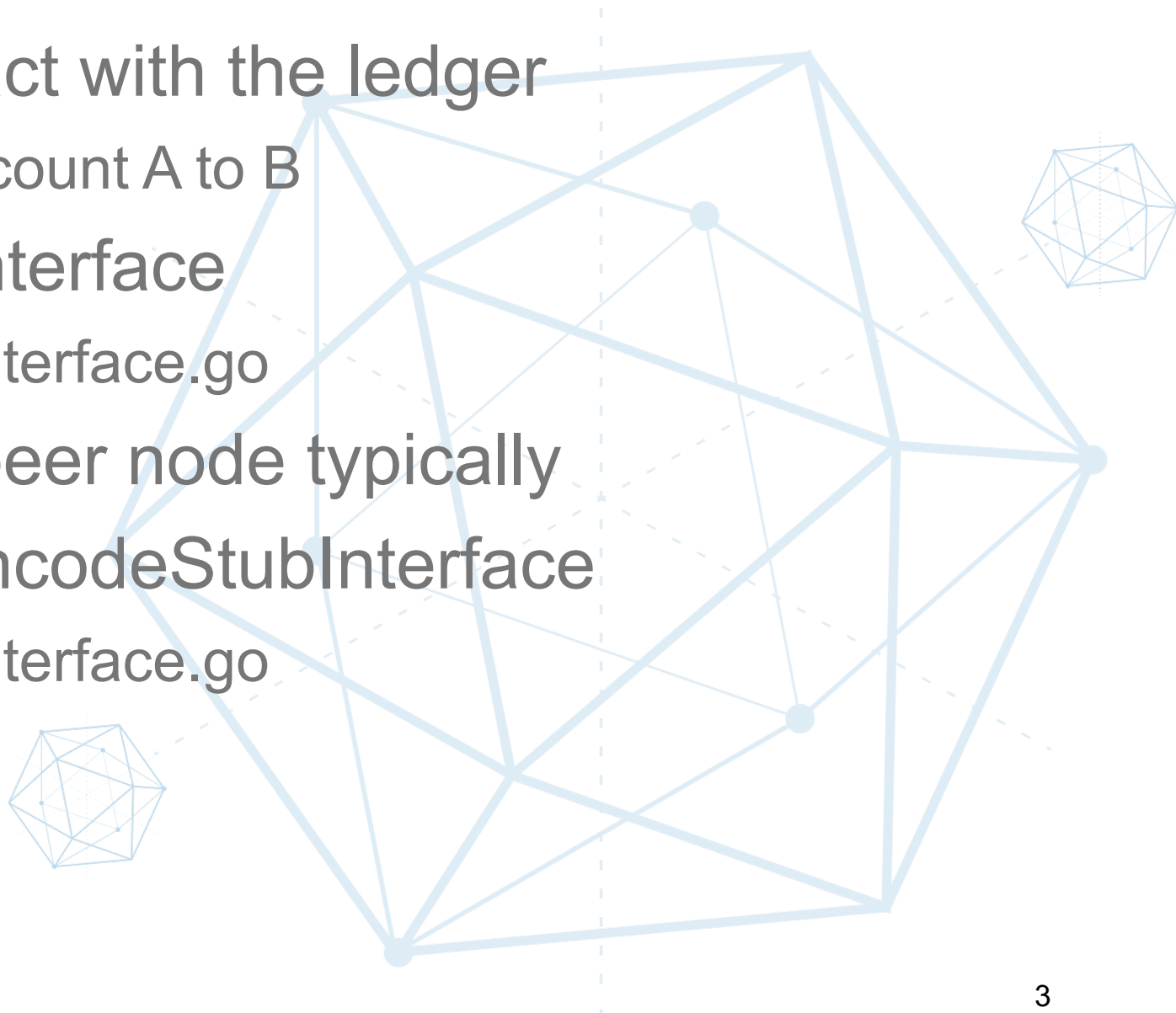
About Me

- **Senior Researcher in IBM**
 - Fintech, Cloud and Analytics
- **Open-Source Contributor**
 - [Hyperledger](#), [OpenStack](#), [OpenDaylight](#), etc.
- **Hyperledger Developer**
 - Core designer & committer of [Fabric](#), [Cello](#), [sdk](#) etc.
 - PTL of [Cello](#) project and [fabric-sdk-py](#) project
 - Chair of [Hyperledger Technical Working Group China](#)



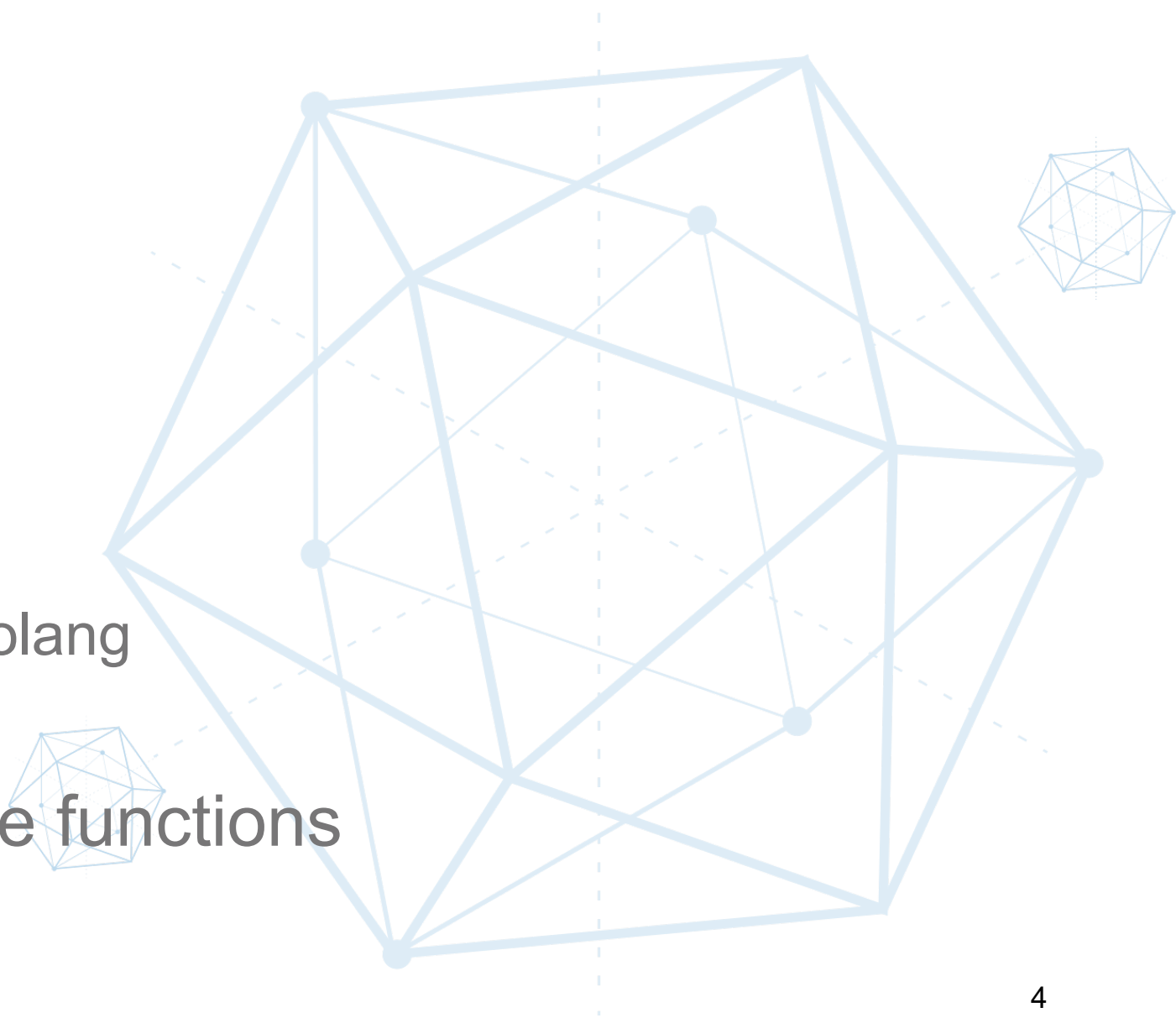
Chaincode Concept

- Application logics to interact with the ledger
 - E.g., transfer money from account A to B
- Implement a Chaincode Interface
 - `fabric/core/chaincode/shim/interface.go`
- Running in containers at peer node typically
- Call Peer's API by a ChaincodeStubInterface
 - `fabric/core/chaincode/shim/interface.go`
- Stateless
- Deterministic



Chaincode Support

- Languages
 - Golang
 - Java
- How to call chaincode
 - CLI
 - SDK: Node, Python, Java, Golang
- Transaction: call chaincode functions



Chaincode Programming

- Implement the Chaincode interface

```
type Chaincode interface {  
    // Init is called during Deploy transaction after the container has been  
    // established, allowing the chaincode to initialize its internal data  
    Init(stub ChaincodeStubInterface) pb.Response  
    // Invoke is called for every Invoke transactions. The chaincode may change  
    // its state variables  
    Invoke(stub ChaincodeStubInterface) pb.Response  
}
```

Demo Chaincode

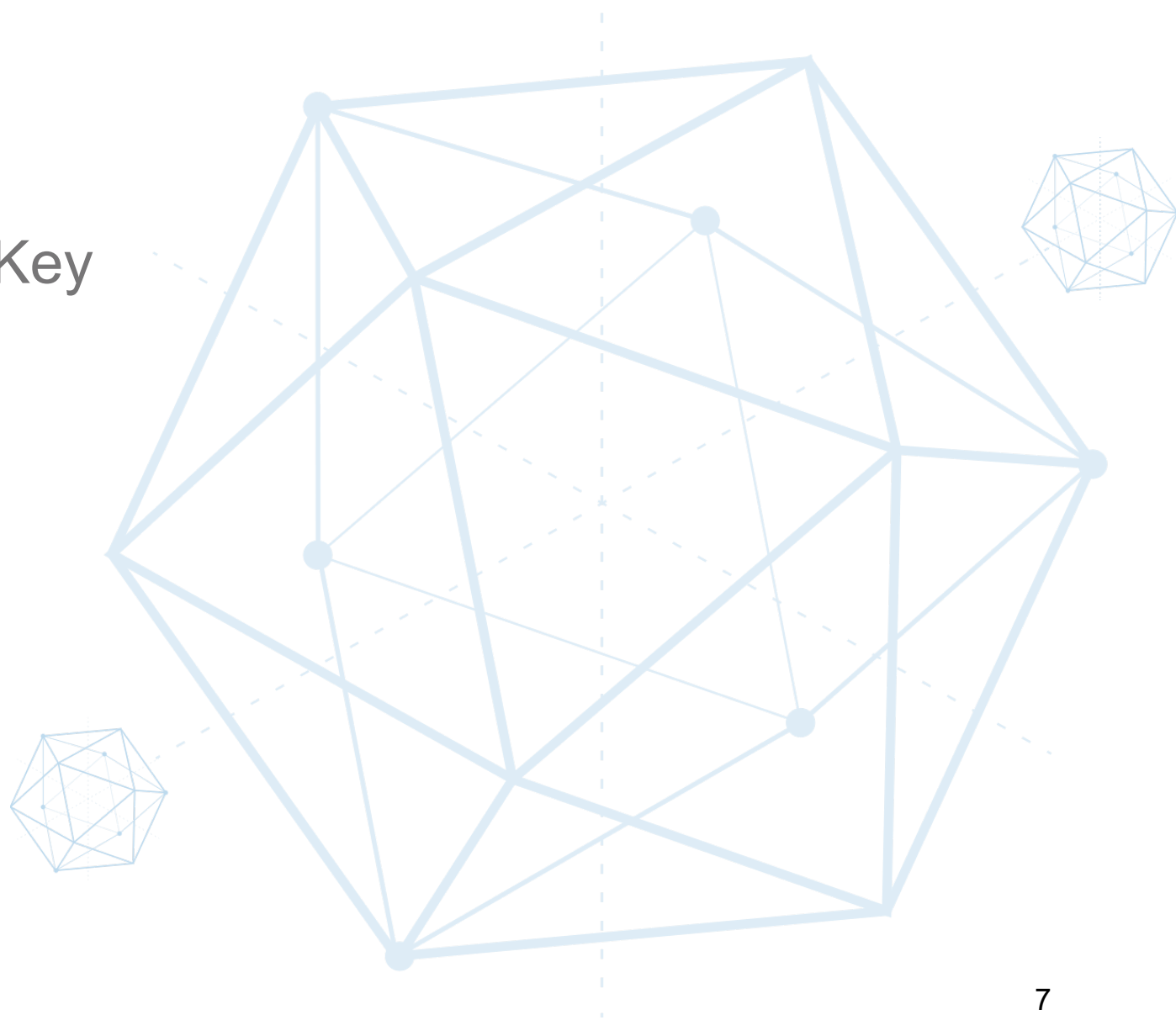
```
package main
import (
    "errors"
    "fmt"
    "github.com/hyperledger/fabric/core/chaincode/shim"
)
type DemoChaincode struct { }
func (t *DemoChaincode) Init(stub shim.ChaincodeStubInterface) pb.Response {
    // more logics using stub here
    return stub.Success(nil)
}

func (t *DemoChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response
    // more logics using stub here
    return stub.Success(nil)
}

func main() {
    err := shim.Start(new(DemoChaincode))
    if err != nil {
        fmt.Printf("Error starting DemoChaincode: %s", err)
    }
}
```

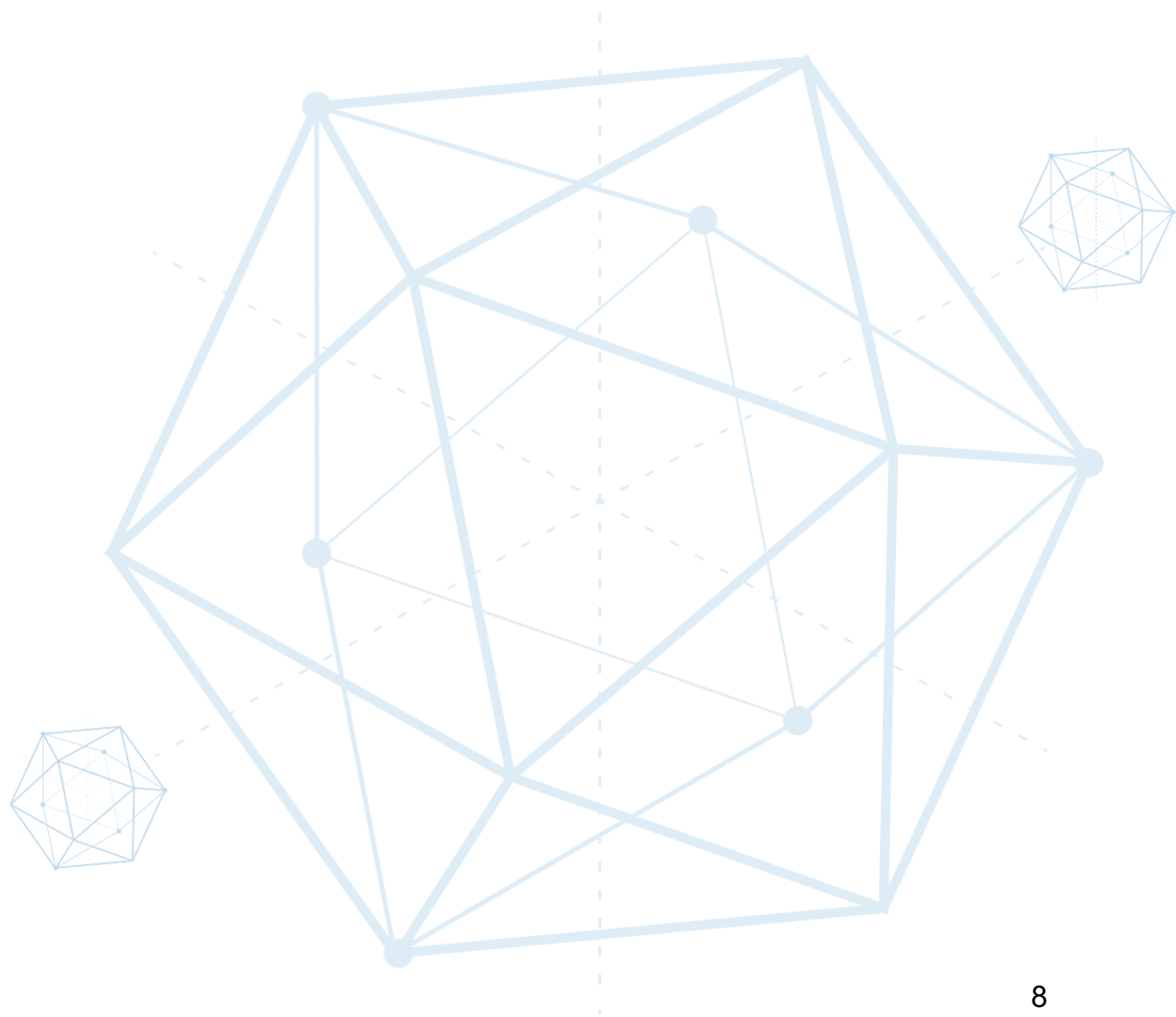
ChaincodeStubInterface

- Ledger State Operations
 - GetState
 - GetStateByPartialCompositeKey
 - GetStateByRange
 - DelState
 - PutState
 - GetHistoryForKey
 - GetQueryResult



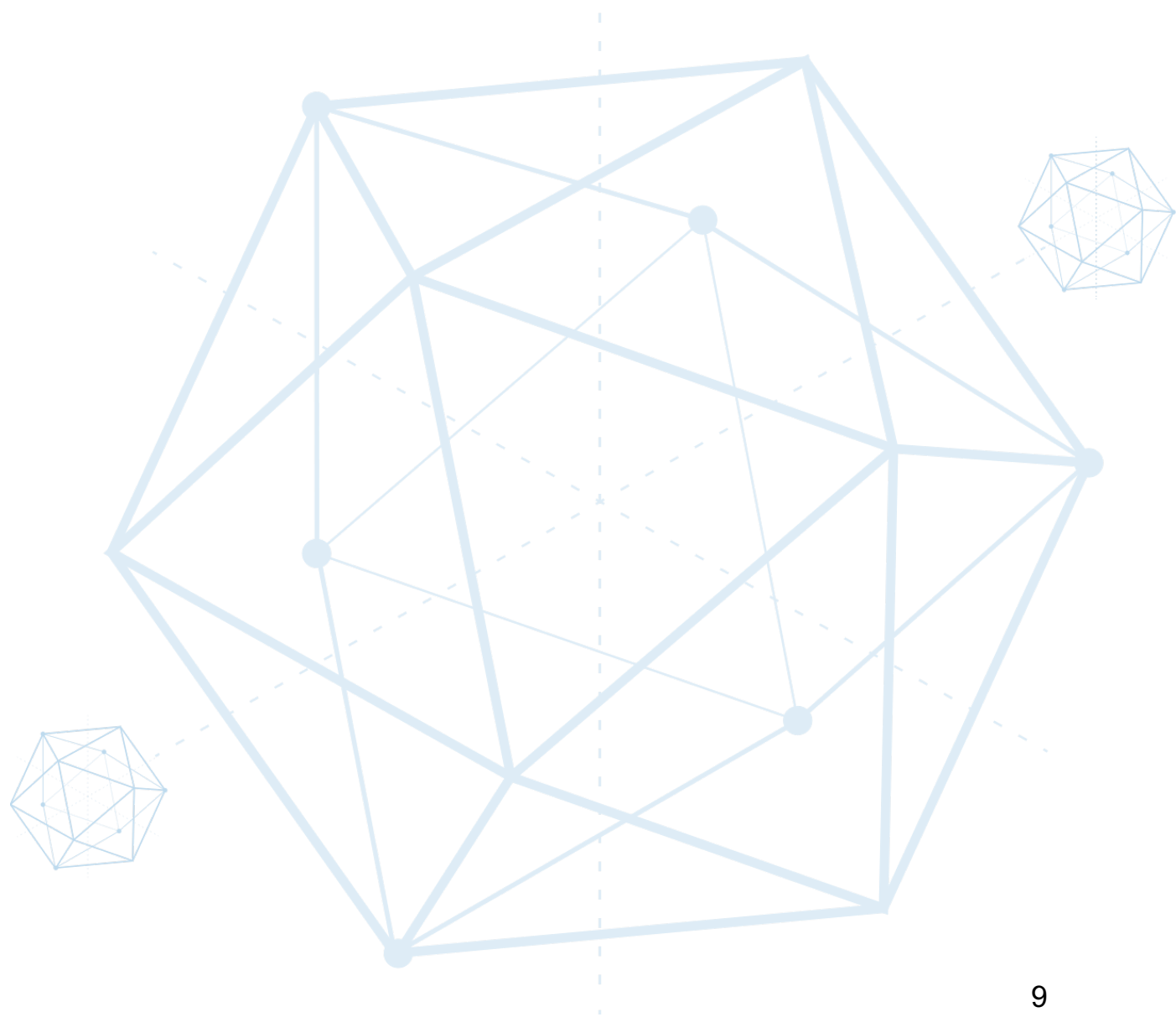
ChaincodeStubInterface

- Transaction Operations
 - GetBinding
 - GetCreator
 - GetSignedProposal
 - GetTransient
 - GetTxID
 - GetTxTimestamp



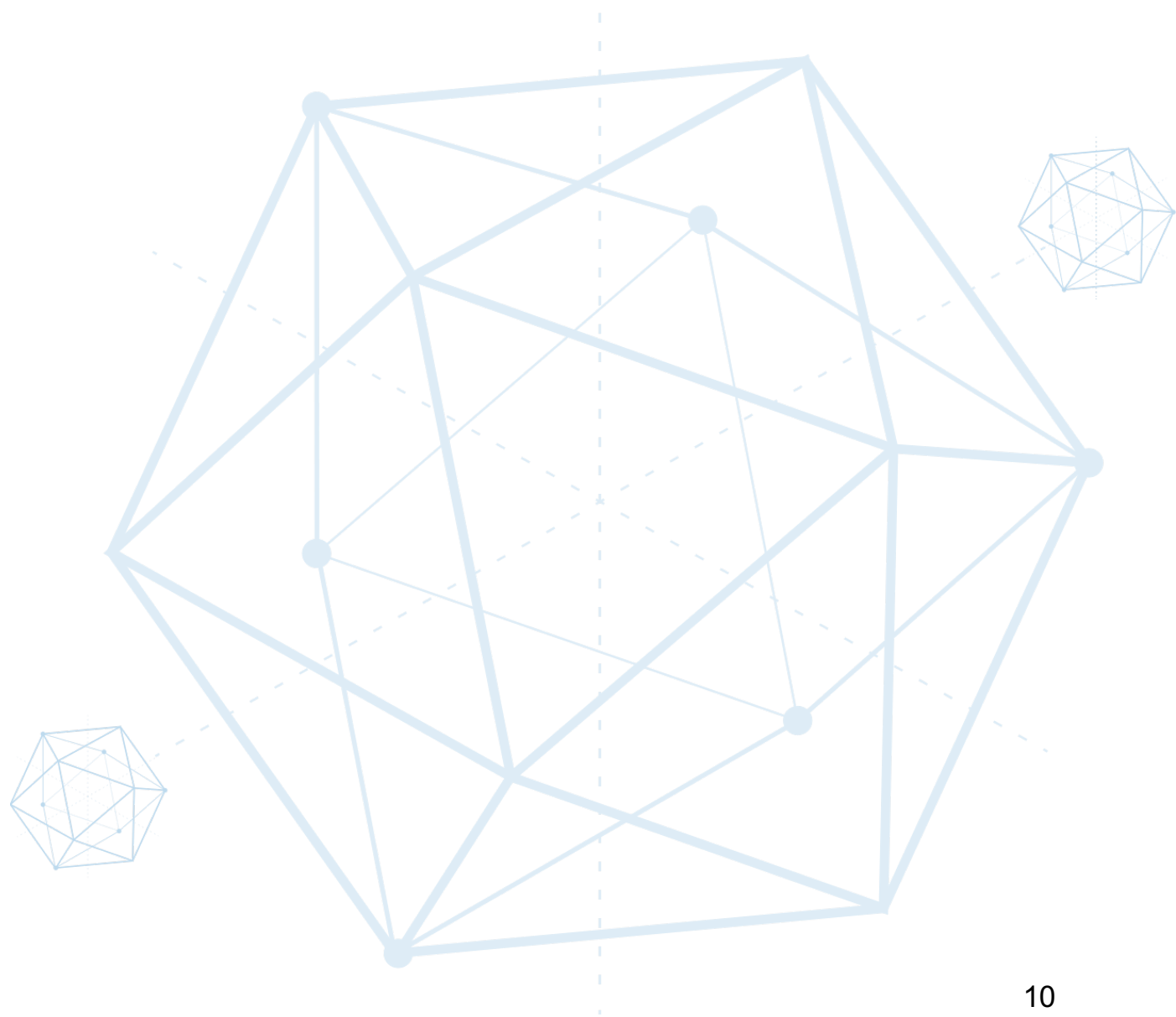
ChaincodeStubInterface

- Stub Arguments
 - GetArgs
 - GetArgsSlice
 - GetFunctionAndParameters
 - GetStringArgs



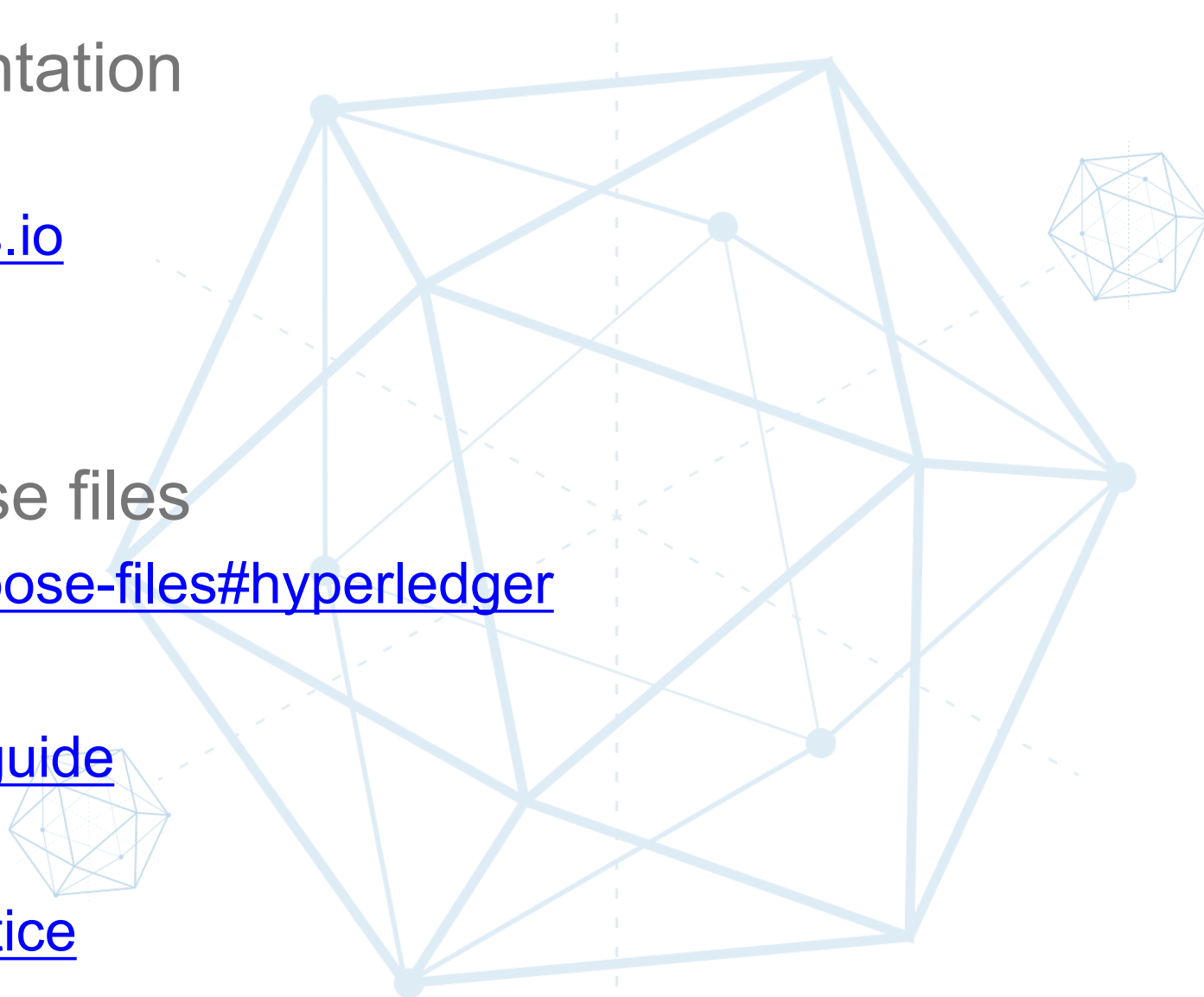
ChaincodeStubInterface

- Others
 - CreateCompositeKey
 - SplitCompositeKey
 - InvokeChaincode
 - SetEvent



Reference

- Hyperledger Wiki&Documentation
 - wiki.hyperledger.org
 - hyperledger-fabric.readthedocs.io
- IBM 区块链
 - ibm.com/ibm/cn/blockchain/
- Hyperledger Fabric Compose files
 - github.com/yeasy/docker-compose-files#hyperledger
- 《区块链技术指南》
 - github.com/yeasy/blockchain_guide
- 《Docker 从入门到实践》
 - github.com/yeasy/docker_practice





HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS



Questions?

Thank You!
@baohua

Slides available at github.com/yeasy/seminar-talk#hyperledger