中山大学数据科学与计算机学院本科生实验报告

(2019 年秋季学期)

课程名称：区块链原理与技术　　任课教师： 郑子彬

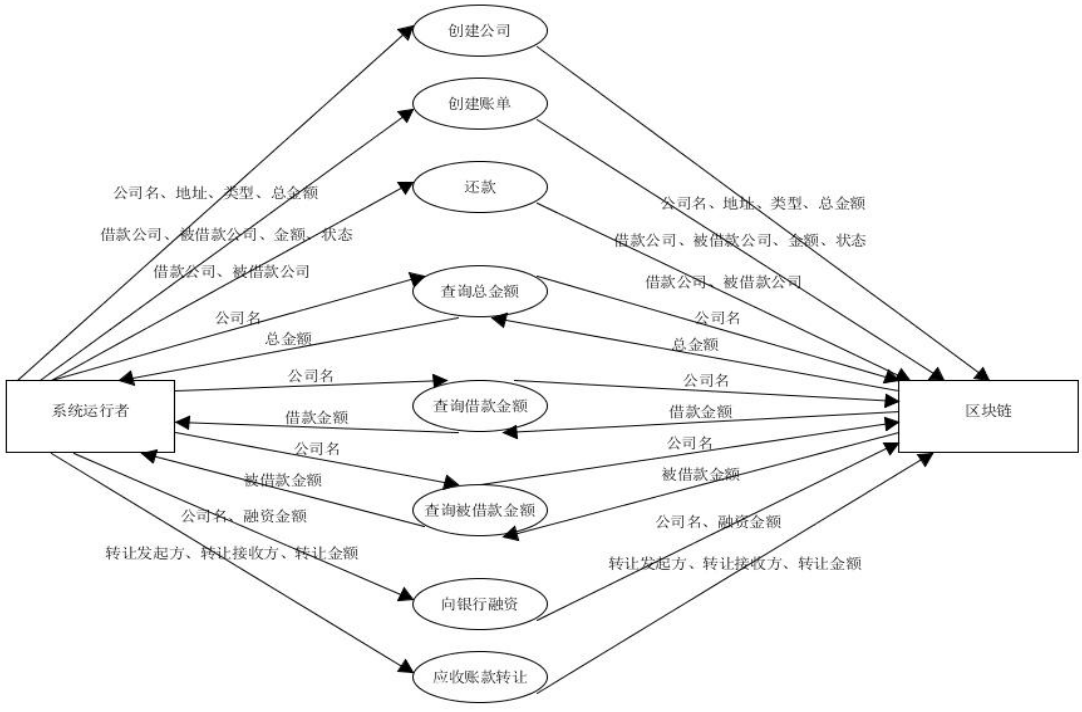
年级	2017 级	专业（方向）	软件工程
学号	17343152	姓名	张凯
电话	13246856351	Email	2112060078@qq.com
开始日期	2019/12/3	完成日期	2019/12/13

一、 项目背景

基于区块链、 智能合约等， 实现基于区块链的供应链金融平台。

二、 方案设计

数据流图:



核心功能介绍:

这次实验完成情况不好，只完成了一个可以在命令行运行的应用。下面看一下核心的功能：

创建公司：

首先看智能合约里的实现：

```
    function createCompany(string n,string a,string c,uint t) public{
        Companies.push(company(n,a,c,t));
     }
```

编译成 Java 之后生成的接口：

```
    public RemoteCall<TransactionReceipt> createCompany(String n, String a, String c,
BigInteger t) {
        final Function function = new Function(
                FUNC_CREATECOMPANY,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(n),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(a),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(c),
                new org.fisco.bcos.web3j.abi.datatypes.generated.Uint256(t)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
}
```

在主类里的调用：

```
        public void myCreateCompany(String n, String a, String c, BigInteger t) {
                try {
            String contractAddress = loadProjectAddr();

            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.createCompany(n,a,c,t).send();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" create company exception, error message is {}", e.getMessage());

            System.out.printf(" create company failed, error message is %s\n",
e.getMessage());
        }
        }
```

创建账单：

智能合约里面的实现：

```
    function createReceipt(string f,string t,uint m,string s){
```

```
        Receipts.push(receipt(f,t,m,s));
}
```

Java 文件生成的接口：

```
    public RemoteCall<TransactionReceipt> createReceipt(String f, String t, BigInteger m,
String s) {
        final Function function = new Function(
                FUNC_CREATERECEIPT,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(f),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(t),
                new org.fisco.bcos.web3j.abi.datatypes.generated.Uint256(m),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(s)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
}
```

在主类里的调用：

```
        public void myCreateReceipt(String f, String t, BigInteger m, String s) {
                try {
            String contractAddress = loadProjectAddr();

            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.createReceipt(f,t,m,s).send();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" create receipt exception, error message is {}", e.getMessage());

            System.out.printf(" create receipt failed, error message is %s\n",
e.getMessage());
        }
        }
```

还款（应收账款结算）：

智能合约里面的实现：

```
    function endReceipt(string f,string t){
        uint i;
        for(i=0;i<Receipts.length;i++){

if(keccak256(Receipts[i].fromCompany)==keccak256(f)&&keccak256(Receipts[i].toCompany)==kec
cak256(t)){
```

```
                uint j;
                for(j=0;j<Companies.length;j++){
                    if(keccak256(Companies[j].name)==keccak256(f)){
                        Companies[j].totalMoney=Companies[j].totalMoney-Receipts[i].money;
                    }
                    if(keccak256(Companies[j].name)==keccak256(t)){
                        Companies[j].totalMoney=Companies[j].totalMoney+Receipts[i].money;
                    }
                }
                Receipts[i].money=0;
            }
        }
}
```

Java 文件生成的接口：

```
    public RemoteCall<TransactionReceipt> endReceipt(String f, String t) {
        final Function function = new Function(
                FUNC_ENDRECEIPT,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(f),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(t)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
    }
```

在主类里的调用：

```
        public void myEndReceipt(String f, String t) {
                try {
            String contractAddress = loadProjectAddr();

            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.endReceipt(f,t).send();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" end receipt exception, error message is {}", e.getMessage());

            System.out.printf("  end  receipt  failed,  error  message  is  %s\n",
e.getMessage());
        }
        }
```

向银行融资：

智能合约里面的实现：

```
    function borrowFromBank(string creator,uint m){
```

```
        uint i;
        uint total=0;
        for(i=0;i<Receipts.length;i++){
            if(keccak256(Receipts[i].toCompany)==keccak256(creator)){
                total=total+Receipts[i].money;
            }
        }
        if(total>=m){
            uint j;
            for(j=0;j<Companies.length;j++){
                if(keccak256(Companies[j].name)==keccak256(creator)){
                    Companies[j].totalMoney=Companies[j].totalMoney+m;
                }
            }
        }
    }
```

Java 文件生成的接口：

```
    public RemoteCall<TransactionReceipt> borrowFromBank(String creator, BigInteger m) {
        final Function function = new Function(
                FUNC_BORROWFROMBANK,
                Arrays.<Type>asList(new
org.fisco.bcos.web3j.abi.datatypes.Utf8String(creator),
                new org.fisco.bcos.web3j.abi.datatypes.generated.Uint256(m)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
}
```

在主类里的调用：

```
        public void myBorrowFromBank(String creator, BigInteger m) {
                try {
            String contractAddress = loadProjectAddr();

            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.borrowFromBank(creator,m).send();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" borrow from bank exception, error message is {}",
e.getMessage());

            System.out.printf("borrow from bank failed, error message is %s\n",
e.getMessage());
        }
        }
```

应收账款转让：

智能合约里面的实现：

```
    function transferReceipt(string creator,string t,uint m){
        uint i;
        uint total=0;
        for(i=0;i<Receipts.length;i++){
            if(keccak256(Receipts[i].toCompany)==keccak256(creator)){
                total=total+Receipts[i].money;
            }
        }
        if(total>=m){
            uint j;
            uint one;
            for(j=0;j<Receipts.length;j++){
                if(keccak256(Receipts[j].toCompany)==keccak256(creator)){
                    one=Receipts[j].money;
                    if(one<m){
                        m=m-one;
                        Receipts[j].money=0;
                        createReceipt(Receipts[j].fromCompany,t,one,"believe");
                    }
                    else{
                        Receipts[j].money=Receipts[j].money-m;
                        createReceipt(Receipts[j].fromCompany,t,m,"believe");
                        break;
                    }
                }
            }
        }
    }
```

Java 文件生成的接口：

```
    public RemoteCall<TransactionReceipt> transferReceipt(String creator, String t,
BigInteger m) {
        final Function function = new Function(
                FUNC_TRANSFERRECEIPT,
                Arrays.<Type>asList(new
org.fisco.bcos.web3j.abi.datatypes.Utf8String(creator),
                new org.fisco.bcos.web3j.abi.datatypes.Utf8String(t),
                new org.fisco.bcos.web3j.abi.datatypes.generated.Uint256(m)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
    }
```

在主类里的调用：

```
        public void myTransferReceipt(String creator, String t, BigInteger m) {
                try {
            String contractAddress = loadProjectAddr();
```

```
            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.transferReceipt(creator,t,m).send();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" transfer receipt exception, error message is {}",
e.getMessage());

            System.out.printf("transfer receipt failed, error message is %s\n",
e.getMessage());
        }
        }
```

查询总金额:

智能合约里面的实现:

```
    function totalMoney(string f) returns(uint){
        uint i;
        for(i=0;i<Companies.length;i++){
            if(keccak256(Companies[i].name)==keccak256(f)){
                emit TotalMoneyEvent(Companies[i].totalMoney,f);
                return Companies[i].totalMoney;
            }
        }
    }
```

Java 文件生成的接口:

```
    public RemoteCall<TransactionReceipt> totalMoney(String f) {
        final Function function = new Function(
                FUNC_TOTALMONEY,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(f)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
    }
```

在主类里的调用:

```
        public void myTotalMoney(String f) {
                try {
            String contractAddress = loadProjectAddr();

            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.totalMoney(f).send();
            List<TotalMoneyEventEventResponse> response =
project.getTotalMoneyEventEvents(receipt);
            if (!response.isEmpty()) {
                            System.out.printf(" The company %s 's total money is %s
```

```
\n", f,
                            response.get(0).ret.toString());
            } else {
                System.out.println(" event log not found, maybe transaction not exec. ");
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error("  get  total  money  exception,  error  message  is  {}",
e.getMessage());

            System.out.printf("  get  total  money  failed,  error  message  is  %s\n",
e.getMessage());
        }
        }
```

查询借款金额:

智能合约里面的实现:

```
    function fromReceipt(string f) returns(uint){
        uint i;
        uint total=0;
        for(i=0;i<Receipts.length;i++){
            if(keccak256(Receipts[i].fromCompany)==keccak256(f)){
                total=total+Receipts[i].money;
            }
        }
        emit FromReceiptEvent(total,f);
        return total;
    }
```

Java 文件生成的接口:

```
    public RemoteCall<TransactionReceipt> fromReceipt(String f) {
        final Function function = new Function(
                FUNC_FROMRECEIPT,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(f)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
    }
```

在主类里的调用:

```
        public void myFromReceipt(String f) {
                try {
            String contractAddress = loadProjectAddr();

            Project  project  =  Project.load(contractAddress,  web3j,  credentials,  new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.fromReceipt(f).send();
```

```
            List<FromReceiptEventEventResponse>                response                =
project.getFromReceiptEventEvents(receipt);
            if (!response.isEmpty()) {
                                System.out.printf(" The company %s 's from receipt is %s
\n", f,
                            response.get(0).ret.toString());
            } else {
                System.out.println(" event log not found, maybe transaction not exec. ");
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error("  get  from  receipt  exception,  error  message  is  {}",
e.getMessage());

            System.out.printf("  get  from  receipt  failed,  error  message  is  %s\n",
e.getMessage());
        }
        }
```

查询被借款金额:

智能合约里面的实现:

```
    function toReceipt(string f) returns(uint){
        uint i;
        uint total=0;
        for(i=0;i<Receipts.length;i++){
            if(keccak256(Receipts[i].toCompany)==keccak256(f)){
                total=total+Receipts[i].money;
            }
        }
        emit ToReceiptEvent(total,f);
        return total;
    }
```

Java 文件生成的接口:

```
    public RemoteCall<TransactionReceipt> toReceipt(String f) {
        final Function function = new Function(
                FUNC_TORECEIPT,
                Arrays.<Type>asList(new org.fisco.bcos.web3j.abi.datatypes.Utf8String(f)),
                Collections.<TypeReference<?>>emptyList());
        return executeRemoteCallTransaction(function);
    }
```

在主类里的调用:

```
        public void myToReceipt(String f) {
                try {
            String contractAddress = loadProjectAddr();
```

```
            Project project = Project.load(contractAddress, web3j, credentials, new
StaticGasProvider(gasPrice, gasLimit));
            TransactionReceipt receipt = project.toReceipt(f).send();
            List<ToReceiptEventEventResponse> response =
project.getToReceiptEventEvents(receipt);
            if (!response.isEmpty()) {
                                System.out.printf(" The company %s 's to receipt is %s \n",
f,
                            response.get(0).ret.toString());
            } else {
                System.out.println(" event log not found, maybe transaction not exec. ");
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
            logger.error(" get to receipt exception, error message is {}", e.getMessage());

            System.out.printf(" get to receipt failed, error message is %s\n",
e.getMessage());
        }
        }
```

以及最后的主函数:

```
    public static void main(String[] args) throws Exception {

        if (args.length < 1) {
            Usage();
        }

        ProjectClient client = new ProjectClient();
        client.initialize();

        switch (args[0]) {
        case "deploy":
            client.deployProjectAndRecordAddr();
            break;
        case "createCompany":
            if (args.length < 5) {
                Usage();
            }
            client.myCreateCompany(args[1],args[2],args[3],new BigInteger(args[4]));
            break;
        case "createReceipt":
            if (args.length < 5) {
                Usage();
            }
            client.myCreateReceipt(args[1],args[2],new BigInteger(args[3]),args[4]);
            break;
        case "transferReceipt":
            if (args.length < 4) {
```

```
            Usage();
        }
        client.myTransferReceipt(args[1], args[2], new BigInteger(args[3]));
        break;
    case "borrowFromBank":
        if (args.length < 3) {
            Usage();
        }
        client.myBorrowFromBank(args[1], new BigInteger(args[2]));
        break;
    case "endReceipt":
        if (args.length < 3) {
            Usage();
        }
        client.myEndReceipt(args[1], args[2]);
        break;
    case "totalMoney":
        if (args.length < 2) {
            Usage();
        }
        client.myTotalMoney(args[1]);
        break;
    case "fromReceipt":
        if (args.length < 2) {
            Usage();
        }
        client.myFromReceipt(args[1]);
        break;
    case "toReceipt":
        if (args.length < 2) {
            Usage();
        }
        client.myToReceipt(args[1]);
        break;
    default: {
        Usage();
    }
    }

    System.exit(0);
}
```

三、 功能测试

测试时项目已经编译完成，编译过程参考官方文档:

- 编译

```
# 切换到项目目录
$ cd ~/asset-app
# 编译项目
$ ./gradlew build
```

因为应用基于 fisco 链，所以要先启动 fisco 节点：

```
fisco-bcos@fiscobcos-VirtualBox:~/fisco$ bash nodes/127.0.0.1/
start_all.sh
try to start node0
try to start node1
try to start node2
try to start node3
 node2 start successfully
 node0 start successfully
 node3 start successfully
 node1 start successfully
```

部署合约：

```
fisco-bcos@fiscobcos-VirtualBox:~/fisco$ cd
fisco-bcos@fiscobcos-VirtualBox:~$ cd asset-app
fisco-bcos@fiscobcos-VirtualBox:~/asset-app$ cd dist
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh deploy
 deploy Project success, contract address is 0xfd34e7e02314e14
bcad4630e2ccf4eee0dba9e35
```

创建几个公司并验证金额是否正确：

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh createCompany "a" "fdsh" "fhj" 1000
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh createCompany "b" "fdsh" "fhj" 500
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh createCompany "c" "fdsh" "fhj" 500
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 500
```

创建两个账单并验证：

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh createReceipt "a" "b" 100 "believe"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh createReceipt "a" "c" 50 "believe"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh fromReceipt "a"
 The company a 's from receipt is 150
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh toReceipt "b"
 The company b 's to receipt is 100
```

可以看到，创建了两个账单：a 借了 b 100，a 借了 c 50，此时，a 的借款为 150，b 的被借款为 100，结果正确。

转让账款并验证：

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh transferReceipt "b" "c" 40
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh toReceipt "b"
 The company b 's to receipt is 60
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh toReceipt "c"
 The company c 's to receipt is 90
```

b 把应收账款转让给 c 40，此时，b 的应收账款现在变成 60，而 c 的应收账款则变成 90，结果正确。

向银行融资并验证：

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 500
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh borrowFromBank "b" 60
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 560
```

b 的初始金额是 500，向银行融资 60，之后变成 560，结果正确。

应收账款结算并验证：

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh endReceipt "a" "b"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "a"
 The company a 's total money is 940
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 620
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh endReceipt "a" "c"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "a"
 The company a 's total money is 850
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "c"
 The company c 's total money is 590
```

a 和 b 之间的借款为 60，结算之后 a 的余额从 1000 变成 940，b 的余额从 560 变成 620。a 和 c 之间的借款为 90，结算之后 a 的余额从 940 变成 850，c 的余额从 500 变成 590 。结果正确。

四、 界面展示

```
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh borrowFromBank "b" 60
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 560
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh endReceipt "a" "b"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "a"
 The company a 's total money is 940
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "b"
 The company b 's total money is 620
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh endReceipt "a" "c"
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "a"
 The company a 's total money is 850
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$ bash asset_r
un.sh totalMoney "c"
 The company c 's total money is 590
fisco-bcos@fiscobcos-VirtualBox:~/asset-app/dist$
```

五、 心得体会

经过这次的实验，我学会了使用 Web3SDK，学会了如何使用生成的 java 合约的接口，还有使用 event 对函数返回结果进行处理，虽然做的不好，不过还是有不小的收获，希望以后能够再接再厉，提高自己的能力。

附：项目结构参考：

```
|-- build.gradle // gradle配置文件
|-- gradle
|   |-- wrapper
|       |-- gradle-wrapper.jar // 用于下载Gradle的相关代码实现
|       |-- gradle-wrapper.properties // wrapper所使用的配置信息，比如gradle的版本等信息
|-- gradlew // Linux或者Unix下用于执行wrapper命令的Shell脚本
|-- gradlew.bat // Windows下用于执行wrapper命令的批处理脚本
|-- src
|   |-- main
|   |   |-- java
|   |         |-- org
|   |             |-- fisco
|   |                   |-- bcos
|   |                         |-- asset
|   |                               |-- client // 放置客户端调用类
|   |                                       |-- AssetClient.java
|   |                               |-- contract // 放置Java合约类
|   |                                       |-- Asset.java
|   |-- test
|       |-- resources // 存放代码资源文件
|           |-- applicationContext.xml // 项目配置文件
|           |-- contract.properties // 存储部署合约地址的文件
|           |-- log4j.properties // 日志配置文件
|           |-- contract //存放solidity约文件
|                   |-- Asset.sol
|                   |-- Table.sol
|
|-- tool
    |-- asset_run.sh // 项目运行脚本
```

合约文件放在 asset-app/src/test/resources/contract

合约编译得到的 java 合约放在 asset-app/src/main/java/org/fisco/bcos/asset/contract

项目的主类放在 asset-app/src/main/java/org/fisco/bcos/asset/client