Week19- 11.14

Day1 Java-Solidity-

The basic types in `solidity` such as the `uint256`, `bytes`, `byte32[]` can be shown in the following.

Solidity Type	web3j.abi type	conflux-Java-sdk Type	Example
uintxxx	Uintxxx	BigInteger	new Uint256(new BigInteger("111"))
address	Address	Address(cfx:xxxx)	new org.web3j.abi.Address(new Address("xxxxx"))
bool	Bool	Boolean	new Bool(true)
string	Utf8String	String	new Utf8String("heyman")
bytesxx	Bytesxx	byte[]	new Bytes32("111".getBytes())
bytes	DynamicBytes	byte[]	new DynamicBytes("111".getBytes())
Static Array (address[2])	StaticArray2		new StaticArray2<>(org.web3j.abi.datatypes.Address.class, xxx)
Dynamic Array (address[])	DynamicArray		new DynamicArray<>(org.web3j.abi.datatypes.Address.class, xxx)

```
struct Foo {
    Address[] addresses;
}
```

```
    public static class BasicAddressesStruct extends DynamicStruct {
        public List<org.web3j.abi.datatypes.Address> addr;

        public BasicAddressesStruct(List<org.web3j.abi.datatypes.Address> addr) {
            super(new
org.web3j.abi.datatypes.DynamicArray<org.web3j.abi.datatypes.Address>(addr));
            this.addr = addr;
        }

        public BasicAddressesStruct(DynamicArray<org.web3j.abi.datatypes.Address> addr) {
            super(addr);
            this.addr = addr.getValue();
        }
    }
```

`bytes[]` , `string` DynamicStruct StaticStruct

Day2 Decode

hexstring hexstring

```
contractCall.callAndGet(Utf8String.class,"text", node, key)
```

java-sdk posRegist `getVotes`

TupleDecoder hexstring decode

```
        BigInteger[] res = new BigInteger[2];

        String rawData = this.call("getVotes", new
Bytes32(Numeric.hexStringToByteArray(identifier))).sendAndGet();
        rawData = Numeric.cleanHexPrefix(rawData);
        TupleDecoder decoder = new TupleDecoder(rawData);
        res[0] = decoder.nextUint256();
```

```
        res[1] = decoder.nextUint256();
```

decoder address bytes decode

web3j FunctionReturnDecoder decode

```
        String rawData = this.call("getVotes", new
Bytes32(Numeric.hexStringToByteArray(identifier))).sendAndGet();

List outputParameters = new ArrayList<TypeReference<Type>>();
outputParameters.add(new TypeReference<Uint256>() {});
outputParameters.add(new TypeReference<Uint256>() {});

List<Type> list = FunctionReturnDecoder.decode(rawData, outputParameters);
```

Day3 decode

```
struct price{
   Uint256 base
   Uint256 premium
}
```

```
    public static class Price extends StaticStruct {
        public BigInteger base;
        public BigInteger premium;

        public Price(Uint256 base, Uint256 premium) {
            super(base,premium);
            this.base = base.getValue();
            this.premium = premium.getValue();
        }
    }
```

price Uint256 StaticStruct DynamicStruct

```
static void decodeStruct() {
  String structData =
"0x0000000000000000000000000000000000000000000000000000000000000020000000000000000000000000000000000000000000000000000000000000006000000000000000000000000000000000000000000000000000000000000000a000000000000000000000000008f03f1a3f10c05e7cccf75c1fd10168e06659be7000000000000000000000000000000000000000000000000000000000000000568656c6c6f000000000000000000000000000000000000000000000000000000000000";
  TypeReference ts = new TypeReference<Price>() {};
  List<Type> list = DefaultFunctionReturnDecoder.decode(structData, Arrays.asList(ts));
  System.out.println(((Price)list.get(0)).base);
}
```

Day4

Web3j `DynamicArray` `StaticArray`

```
//struct BasicAddresses{
//address[] addr;
//}

String hash = account.call(opt, contract_address, "write", basicAddresses);
```

`MethodSignature` signature methodid encode rawdata

, methodid methodid rawdata

web3j DefaultFunctionEncoder.java encodeFunction

```
    @Override
    public String encodeFunction(final Function function, String signature) {
      String methodId = buildMethodId(signature);
        final StringBuilder result = new StringBuilder(methodId);

        return encodeParameters(parameters, result);
    }
// methodSignature ( ) (xxx) writeBasicAddressesStruct((address[]))

    public static String buildMethodId(final String methodSignature) {
        final byte[] input = methodSignature.getBytes();
        final byte[] hash = Hash.sha3(input);
```

```
        return Numeric.toHexString(hash).substring(0, 10);
    }
```

Function

```
        Function f = new Function(
                "writeBasicAddressesStruct",
                Arrays.asList(struct),
                Collections.emptyList()
        );
```

rawdata

```
String hash = acc.callWithData(new Address(addr), t);
```

Day5 web3j

web3j web3j cc `java.lang.UnsupportedOperationException: Array types must be wrapped in a TypeReference`

https://github.com/web3j/web3j/issues/1726

web3j bug

Revision #14
Created 14 November 2022 09:01:01 by Xianqi
Updated 18 November 2022 12:45:49 by Xianqi