LIBRARY

Or Library Driven Development

WHAT WE DO AND WHAT NOT

- Highlights a few parts from the solidity docs
- Is not a replacement for the Documentation
- In our course we will no directly use any libraries
- But they can be an important part in a development workflow

WHAT IS A LIBRARY?

- A smart contract
- On an address
- A set of re-usable functions

- Does not have a storage
- Cannot hold ether

- Re-usage doesn‘t contaminate the blockchain

MORE

- Standard Library doesn‘t exist currently
  - But many are in development
  - OpenZeppelin for example
- Could be audited
  - And should be
- Uses DELEGATECALL
  - Delegates the whole context
  - Looks for the “library” as if it was the contract

LIBRARIES

- Starts with `library C {}`

- “Libraries can be seen as implicit base contracts of the contracts that use them”

LINKING

- Not like inheritance
  - `contract myContract is anotherContract { }`
- Happens at bytecode level
  - Compilation leaves a placeholder for the library address
  - Without filling in the address, the bytecode is invalid
- Linking is a simple „replace" operation with the address of the library
- Good Writeup: https://ethereum.stackexchange.com/questions/6927/what-are-the-steps-to-compile-and-deploy-a-library-in-solidity

LINKING

- Compilation
  - solc --optimize --bin yourContract.sol
- Binary 60606040….__LibraryName_____123
- Linking replaces __LibraryName_____ with address
  - solc --optimize --bin yourContract.sol | solc --link --libraries LibraryName:<address>

UPGRADABILITY

- Not directly
  - Can't upgrade code on an address
- But people are working on solution
  - Can upgrade address of the library
  - With Proxies
  - https://openzeppelin.org

STORAGE

- Any modification the library will be saved in the contracts own storage
- Think of it as a pointer (from C, C++) to another function.
- `function insert(Data storage self, uint value)`
  - First parameter in a library function with “storeage” keyword
  - Reference instead of a copy
  - From Solidity-Docs example http://solidity.readthedocs.io/en/develop/contracts.html#libraries

EVENTS

- Libraries can fire events
- Will be shown as the „contracts‘ event“, not the libraries

- Problematic, because not in the ABI

- Workaround: Place the Event-Definition in both, the library and the contract.

FUNCTION OVERLOADING

- using A for B;
  - using BigInt for uint;
  - Will call first the BigInt library

  - uint myNumber;
  - myNumber.increase();

FUNCTION OVERLOADING

```
//example from the solidity docs!
pragma solidity ^0.4.0;

library Search {
   function indexOf(uint[] storage self, uint value) returns (uint) {
      for (uint i = 0; i < self.length; i++)
         if (self[i] == value) return i;
      return uint(-1);
   }
}
```

FUNCTION OVERLOADING

```
contract C {
    using Search for uint[];
    uint[] data;

    function append(uint value) {
        data.push(value);
    }

    function replace(uint _old, uint _new) {
        // This performs the library function call
        uint index = data.indexOf(_old);
        if (index == uint(-1))
            data.push(_new);
        else
            data[index] = _new;
    }
}
```

YOUR TAKE AWAY

- Libraries are great
  - But there are virtually no standard-libs quite yet
- Libraries are not inherited, they are like a part of the contract
- Function overloading can be practical
- People are working on upgradable code (basically re-linking)

THANKS

- Questions?
  - No problem! Head over to the Q&A
- Missing some coding samples?
  - Look at our repository
  - The docs are great!
- Feedback?
  - We'd be happy if you have some.