Blockchain Bootcamp Day 10 (40 minutes)

Understanding Gas and Gas Costs Step By Step

* Gas Calculation
* Three fields are important, gas, transaction cost, and execution cost.
* Debug mode of transaction, in the step detail indicates the cost of each step.
* How does execution cost differ from transaction cost.
* Each transaction has base fee.
* How much data is sent indicates how much a execution costs.
* 21000 gas paid for each transaction.
* In the whitepaper it will contain costs of each operation.
* If you start a transaction on mainnet, specify gwei per cost. Such as 1 gwei or 100 gwei.
* If many people start transactions at the same time, then they are many transactions in the transaction pool.
* 15 transactions per 10 or 20 seconds can be processed.
* How much gas is limited, loops can spend a lot of gas. Not use loop, except if you know how many times it would loop.

Libraries and using for keyword

* Libraries can used for multiple purpose such as check if the amount is greater than current balance to prevent wrap arounds.
* Zeppelin project has a lot of smart contracts that can be reused.(openzeppelin-contracts)
* Can import libraries similar to importing contracts to derive from, then use it by the **using** keyword -> this will give access functions in the contract and based on the datatype set. To use it on data type itself, instead of referencing library.
* Libraries – How to extend smart contract
* Typical library properties, similar written as smart contract
* They are executed in the context of calling contract.
* The evm is calling another smart contract.
* Libraries are considered stateless
* No state variables
* No ether can be send to a library.
* Libraries are a tool set of functions.
* All these quirks to be lifted in the future possibly.
* Libraries are executed a little bit like copying the code over to your contract.
* Libraries are not to upgrade smart contracts.
* It’s better to have logic stored in another smart contract that you can upgrade.
* Storage in one smart contract, logic in another smart contact to upgrade.

Solidity Version 6 breaking changes.

* Functions can only be overridden using virtual, especially outside interface
* Byte code changes slightly compared to a function without virtual keyword
* Override needs to specify order it’s overriding, it doesn’t’ matter how it’s order as long the derived class is following the right order.
* State variable shadowing is not allowed.
* Receive function(using receive keyword) and fallback function declared using **fallback keyword** will replace fallback function(or a function without a name)
* Try/catch can catch errors from external function calls. Multiple catch blocks, throwing Error and LowLevelErrors