

Lecture 14 - Start on Solidity

News

1. Ethereum successful upgrade to Constantinople <https://decryptmedia.com/5445/ethereum-hard-fork-constantinople-upgrade>
2. Banks moving operations out of US due to KYI

Solidity

Version of the compiler

```
pragma solidity ^0.4.24;
```

Library we will be using

```
import "openzeppelin-solidity/contracts/token/ERC20/StandardToken.sol";
```

Comments

```
/* Some Comment */ // Single line comment
```

Classes and Inheritance

```
contract StandardToken is ERC20, BasicToken {
```

Constructor

```
    constructor() public {
```

Declare a string

```
string public constant name = "SimpleToken"; // solium-disable-line uppercase
```

Test for true statments

```
require(_to != address(0));
```

if

```
if (_subtractedValue > oldValue) {  
    allowed[msg.sender][_spender] = 0;  
} else {  
    allowed[msg.sender][_spender] = oldValue.sub(_subtractedValue);  
}
```

Function

```
function transferFrom( address _from, address _to, uint256 _value  
) public returns (bool)  
{
```

Events

```
event Approval( address indexed owner, address indexed spender,  
    uint256 value );
```

```
emit Approval(msg.sender, _spender, _value);
```

Example with a Loop

```
1 pragma solidity ^0.4.24;
2
3 import "openzeppelin-solidity/contracts/ownership/Ownable.sol";
4
5 contract Loop is Ownable {
6     struct StudentStruct {
7         uint256 grade;          // grade is a percentage * 1000
8         uint256 pin;            // Unique PIN code for student to get grade
9     }
10
11     mapping(address => StudentStruct) addressToStudent;
12     address [] studentList;
13
14     event LogStudentGrade(address student, uint256 studentGrade);
15
16     function appendStudentGrade(address student, uint256 studentGrade,
17     uint256 pin) public onlyOwner {
18         studentList.push(student);
19         addressToStudent[student].grade = studentGrade;
20         addressToStudent[student].pin = pin;
21     }
22
23     function getStudentCount() public view returns(uint) {
24         return studentList.length;
25     }
26
27     function getNthStudentGrade(uint i)
28     public view onlyOwner returns(uint256) {
29         require(i >= 0);
30         require(i < studentList.length);
31         return(addressToStudent[studentList[i]].grade);
32     }
33
34     function getNthStudentAddress(uint i)
35     public view onlyOwner returns(address) {
36         require(i >= 0);
37         require(i < studentList.length);
38         return(studentList[i]);
39     }
40 }
```

```
37
38  function getAGrade(address student, uint256 pin)
    public view returns(uint256) {
39      require(msg.sender == address);
40      require(pin == addressToStudent[student].pin);
41      return(addressToStudent[student].grade);
42  }
43
44  function emitGrades () public onlyOwner {
45      for (uint i=0; i<studentList.length; i++) {
46          emit LogStudentGrade(studentList[i],
              addressToStudent[studentList[i]].grade);
47      }
48  }
49 }
```