

Solidity tutorial

Joseph Bonneau

FOSAD 2017

Bertinoro, Italy

Schedule

Monday: Intro to Ethereum

- Ethereum system design
- Overview of solidity
- Overview of applications

Tuesday: Practical Exercise

- Solidity coding tutorial

Wednesday: Security issues for Ethereum Development

- Secure smart contract programming
- Classic security bugs: reentrancy, unchecked sends
- DAO case study

Solidity

Solidity should look familiar

- Syntax looks like C++, JavaScript etc.
- Contracts look like classes/objects
 - Can mark functions `internal`
- Static typing
 - Most types can be cast e.g. `bool(x)`

Solidity types

- `bool, uint8, uint16, ... uint256, int8, ... int256`
- `address`
- `string`
- `byte[]`
- `mapping(keyType ==> valueType)`

Clever implementation of maps in Solidity

```
mapping(string => uint256) balances;
```

Alice	15
Bob	15
Joe	100



- every item requires at least one 256-bit word
- `balances["Andrew"]` is 0 if "Andrew" doesn't exist or if "Andrew" has 0 balance
- to delete a key, set `balances["Andrew"] = 0`
- Cannot delete an entire map!

Polite contracts call `revert()` on errors

```
uint8 numCandidates;
uint32 votingFee;
mapping(address => bool) hasVoted;
mapping(uint8 => uint32) numVotes;

/// Cast a vote for a designated candidate
function castVote(uint8 candidate) {
    if (msg.value < votingFee)
        return;
    if (hasVoted[msg.sender])
        revert();

    hasVoted[msg.sender] = true;
    numVotes[candidate] += 1;
}
```

`revert()` ensures no effects persisted except gas consumption

Polite contracts call `selfdestruct()`

```
/// Cast a vote for a designated candidate  
function finishElection() {  
    selfdestruct(creator);  
}
```

`creator` receives remaining funds

Modifiers ease repetitive safety checks

```
address public owner;  
uint public electionEnd;
```

```
modifier onlyBy(address _account){  
    require(msg.sender == _account);  
    _;  
}
```

```
modifier onlyAfter(uint _block) {  
    require(block.blocknumber >= _block);  
    _;  
}
```

```
function endElection()  
    onlyBy(owner) onlyAfter(electionEnd){
```

Built-in support for calling other contracts

- `a.send(x)` sends `x` to address `a`
 - returns 0 if this fails due to call stack
- `foo.call.value(3).gas(20764)(bytes4(sha3("bar()")));`
 - also `callcode`, `delegatecall`
 - default is 0 value, all available gas
- `new` constructor deploys a new contract
 - Careful, it's expensive!

Remember:

Smart contracts code is fixed *forever*.
Calls required to update functionality

Solidity gotchas (many more tomorrow)

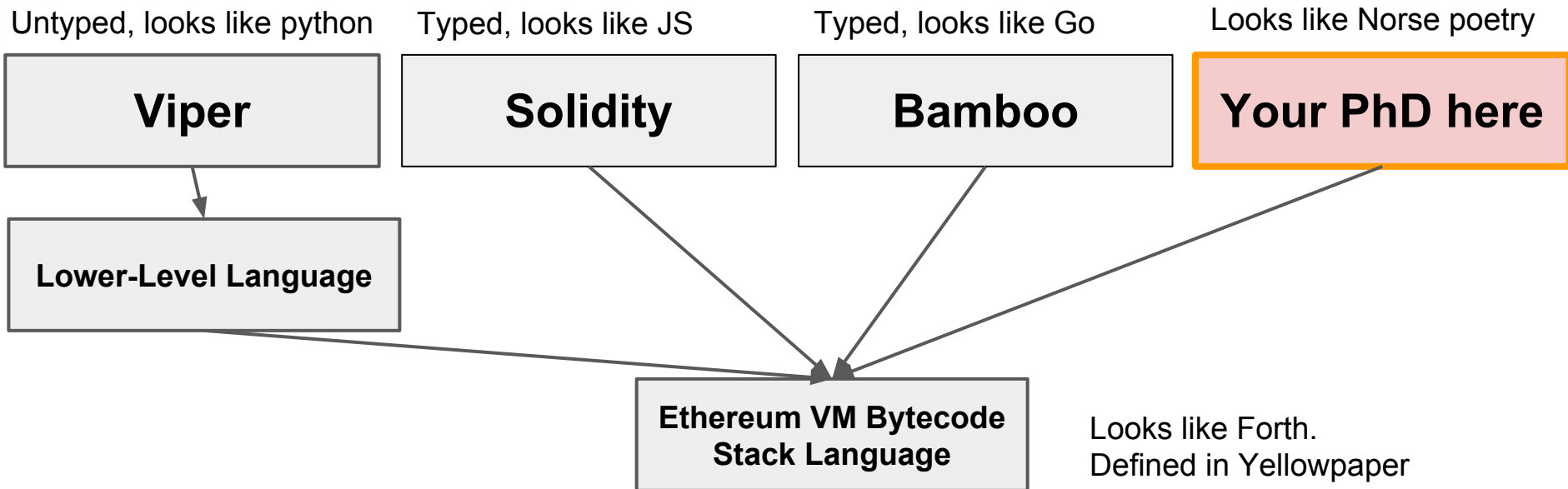
- Member variables `public` by default
 - Setters, getters automatically provided
- Functions must be marked `payable` to accept funds
- Member variables go to storage by default
 - Method variables go to memory
- Fallback function `()`
 - Called if no function specified (e.g. `send`)
 - Called if non-existent function called
- `msg.sender` vs. `tx.origin`

Solidity and EVM may outgrow Ethereum itself



- Enterprise Ethereum Alliance, still in infancy (Announced Feb 28)
- Goal: support EVM, Solidity and tools for private blockchains
 - maintain compatibility with Ethereum network

Don't like Solidity? Write your own language!



Exercises

Instructions: goo.gl/Gp2dyw

Code: goo.gl/3ACrVr