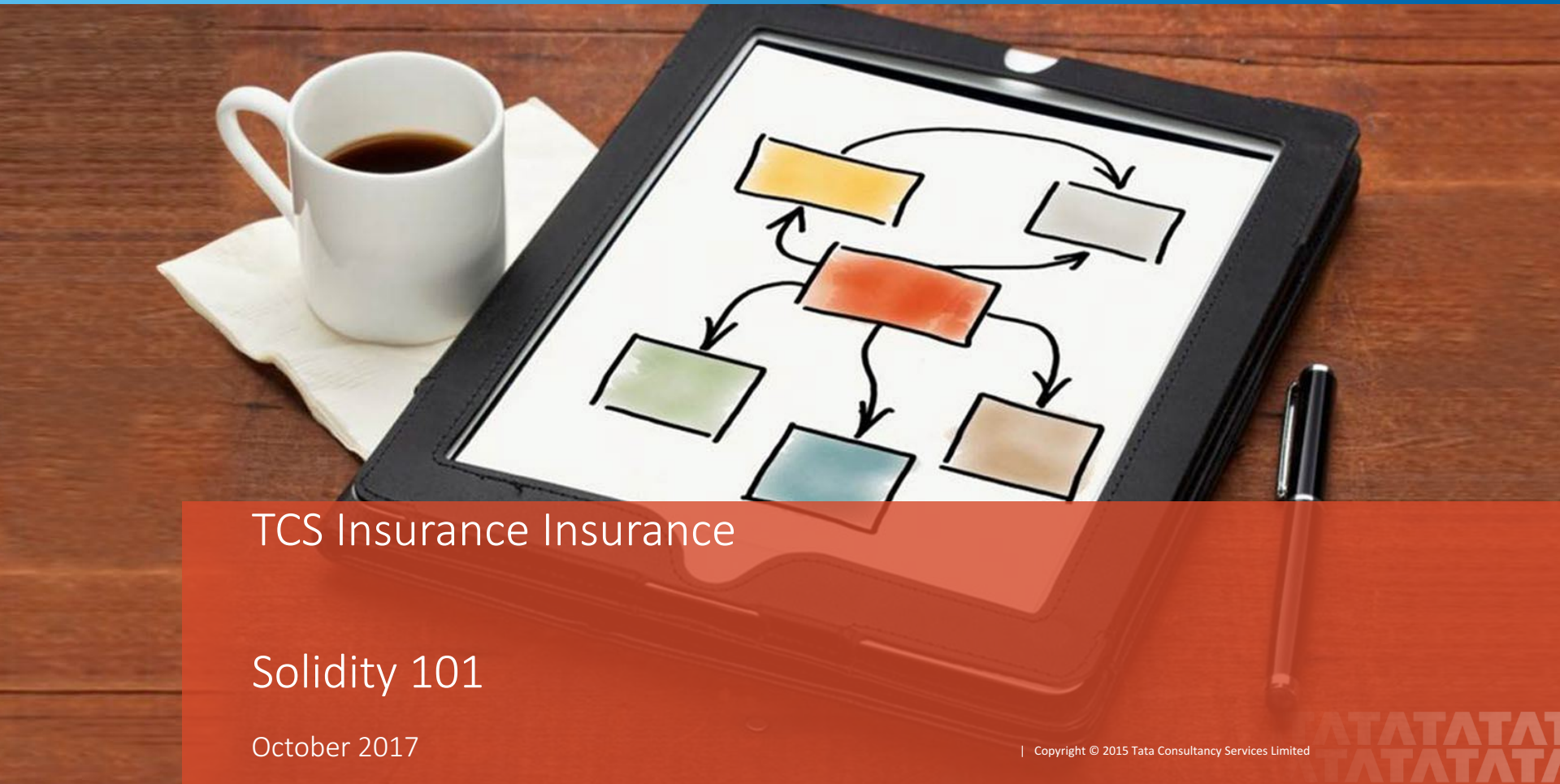


TCS Insurance Insurance

Solidity 101

October 2017

| Copyright © 2015 Tata Consultancy Services Limited

Solidity 101 Workshop : Objective

Basic concepts – **JUST enough** for you to write your FIRST contract

You will **NOT** become an EXPERT after the hands on session

Our objective – help you get started with Blockhain Development

TCS Confidential

Solidity 101

Comparison with JAVA

Layout of a contract

Memory management

Basic Data Types

Arrays

String & bytes

Mapping

Functions

Constants Functions

Constant Variables

Functions

Global Vars | Funcs

Events

TCS Confidential

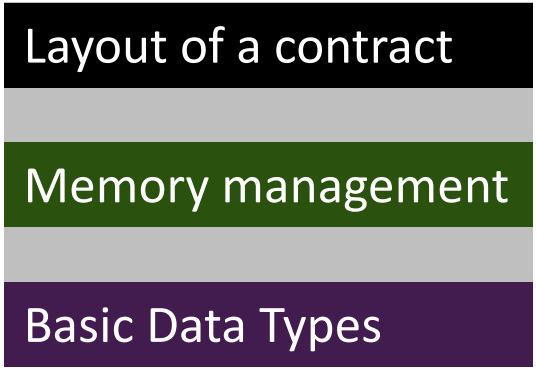
Java Versus Solidity

- Both support object orientated design but there are some differences
  - Single constructor
  - Multiple inheritance

- Both are statically typed, support libraries, and user-defined types
- However, Solidity is **DETERMINISTIC** & single-threaded

A Smart Contract is **LIKE** a JAVA Class

A deployed Contract is **LIKE** a Object instance in JVM


Layout of a contract
Memory management
Basic Data Types

Contract Layout

pragma

contract

function

```
pragma solidity ^0.4.4;

contract HelloEthereum {

  // Storage variable for holding last caller
  // Declaring a storage variable public leads to
  // Automatic generation of getter lastCallerName()
  bytes32 public  lastCallerName;

  // Storage variable for holding the
  mapping(address => bytes32)  addressNames;

  // Event gets emitted everytime someone calls the he
  event HelloInvoked(bytes32 indexed name);

  function HelloEthereum() {
    // constructor
    lastCallerName = 'not-set';
  }

}
```

Storage variable declarations

event

Contract Constructor

- Unlike JAVA only 1 constructor is allowed
  - Keyword function need to be used
  - Name = Same as the contract

TCS Confidential

Runtime Memory Usage

Storage:

- State variables
- Persisted on disk/ledger

Memory:

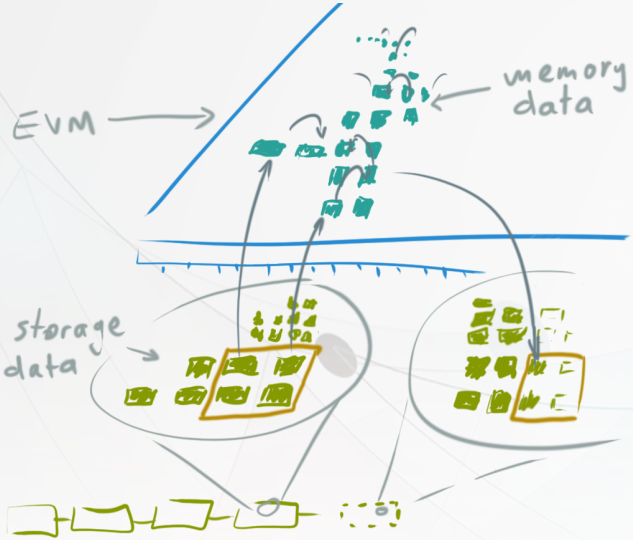
- Local variables in function

Calldata

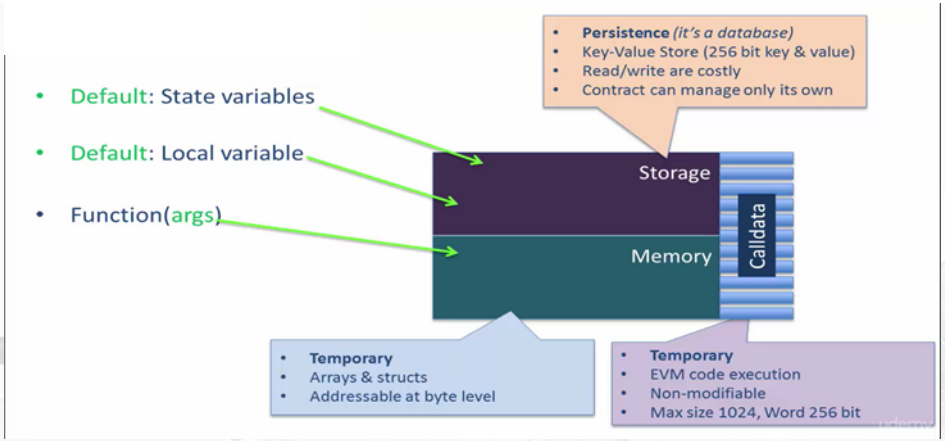
- Function arguments
- Parameters (not return) of external functions

TCS Confidential

Data Location



MOVE TO End/Remove



LOGS

https://pospi.github.io/talk-solidity-blockchain-intro/#18

TCS Confidential

Basic Types

boolean:

- true false
- !, && , ||, ==, !=

integer:

- int and uint
- signed or unsigned
- Size specific as suffix e.g., uint8

byte:

- An array of 1 byte

TCS Confidential

Storage variable visibility

- By default internal

public

- Compiler automatically generates a (getter) function with the same name as the variable

TCS Confidential

Types – Address

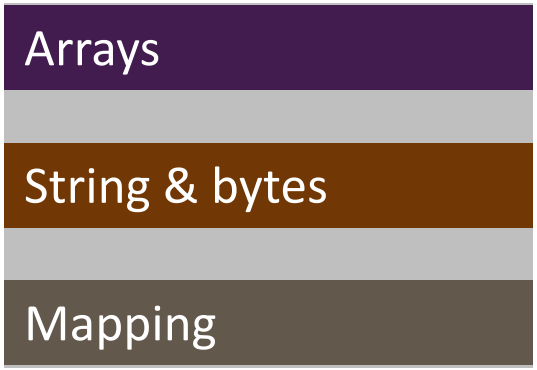
- 20 Bytes
- Stores addresses of *contracts* or *Externally Owned Account*

```
address x = 0x123;
address myAddress = this;
if (x.balance < 10 && myAddress.balance >= 10) x.transfer(10);
```

Multiple functions:

My.Address.**balance(…)** My.Address.**send (…)**

TCS Confidential


Arrays
String & bytes
Mapping

Arrays

Creating a dynamic array:

- Members: **push**
- Can be resized in storage
- *BUT not in memory*

Creating a fixed sized array:

- Members: **length**

TCS Confidential

Special Arrays

- bytes: Dynamically-sized byte array
  - A bytes is similar to byte[], but there are difference
- string: Dynamically-sized UTF-8-encoded string
  - string is similar to bytes but it does not allow indexed access

TCS Confidential

Fixed Length byte Array

Created by suffixing number after bytes

bytes8

bytes32

- byte = bytes1

TCS Confidential

Mapping Type

mapping(_KeyType => _ValueType)

- Similar to JAVA Hashtable
- No concept of NULL in Solidity
- mapping returns a 0-value for non existent keys

TCS Confidential

Mapping Type

```
// Storage variable for holding the address name pair
mapping(address => bytes32)  addressNames;
```

```
function  hello(bytes32 name){
  // Invoke the event
  HelloInvoked(name);

  // Add to mapping
  addressNames[msg.sender] = name;

  // Set the last caller name
  lastCallerName = name;
}
```


TCS Confidential

Struct type

- Complex type
  - Add to mapping
  - Can be nested

```
pragma solidity ^0.4.0;

contract Ballot {

    struct Voter { // Struct
        uint weight;
        bool voted;
        address delegate;
        uint vote;
    }
}
```


TCS Confidential


Functions
Constants
Getters & Setters

Functions

```
uint myAge;
bool myGender;

//returns the tuple containing myAge and myGender
function getMyData returns (uint, bool)
{
    return (myAge, myGender);
}

function setMyAge(uint age)
{
    //so this sets myAge = age, and doesn't change myGender
    getMyData() = (age,);
}

function setMyGender(bool gender)
{
    getMyData() = (,gender);
}
```

Invoked from inside & outside the contract

Changes the state of the contract

Visibility controlled by developer

TCS Confidential

Function Visibility

Only calls from outside the EVM allowed

- External,

Part of the ABI – Interface definition

- Public,

- Internal, Private

Cannot be invoked from outside the EVM ; not part of ABI

TCS Confidential

Functions Arguments

- Internal functions all `struct` as argument *BUT*
- External & Public function CANNOT have `struct` as argument
  - Pass individual elements of struct as arguments

TCS Confidential

Functions Returns

Function can return multiple values

- tuple type : a list of objects of potentially different types whose size is a known at compile-time

- Cannot return struct

TCS Confidential

Constant Function

```
pragma solidity ^0.4.0;

contract HelloWorldContract {

string  word = "Hello World";

function getword() constant returns(string){
    return word;
}

}
```

Does NOT change the state of contract

Caller pays 0 ETH to invoke constant function

As data is retrieved from local node

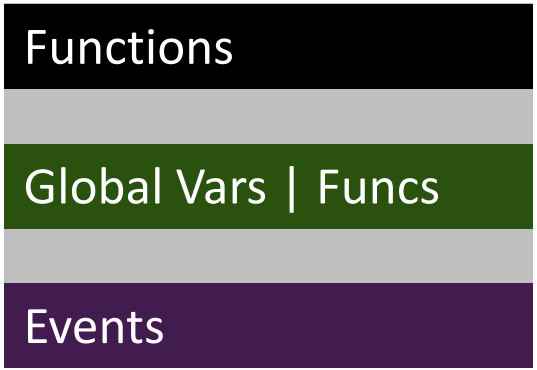
TCS Confidential

Constant Variable

```
string constant myName = "Spencer";
```

- Initialized as part of declaration
- Unlike JAVA – cannot be initialized in constructor

TCS Confidential


Functions
Global Vars | Funcs
Events

Global Objects = block, msg, tx

All of these global objects are available in functions

- `block.blockhash(uint blockNumber) returns (bytes32)`: hash of the given block - only works for 256 most recent blocks excluding current
- `block.coinbase` (`address`): current block miner's address
- `block.difficulty` (`uint`): current block difficulty
- `block.gaslimit` (`uint`): current block gaslimit
- `block.number` (`uint`): current block number
- `block.timestamp` (`uint`): current block timestamp as seconds since unix epoch
- `msg.data` (`bytes`): complete calldata
- `msg.gas` (`uint`): remaining gas
- `msg.sender` (`address`): sender of the message (current call)
- `msg.sig` (`bytes4`): first four bytes of the calldata (i.e. function identifier)
- `msg.value` (`uint`): number of wei sent with the message
- `now` (`uint`): current block timestamp (alias for `block.timestamp`)
- `tx.gasprice` (`uint`): gas price of the transaction
- `tx.origin` (`address`): sender of the transaction (full call chain)

TCS Confidential

Utility Global Functions

Ether units

- convert between the subdenominations of Ether – wei, finney, szabo, ether

Time units

- seconds, minutes, hours, days, weeks and years

Mathematical and Cryptographic functions

- Hashing functions, ecrecover (signature verification)

TCS Confidential

msg : Message Object

- msg is global object that is available in the context of a transaction

msg.sender = Address of the caller

```
//person or account who deployed this contract
address chairperson = msg.sender;
```


TCS Confidential

Contracts can emit events

Declare

```
event HelloInvoked(string indexed name);
```

Emit

```
function  hello(string name){
  // Invoke the event
  HelloInvoked(name);

  // Add to mapping
  addressNames[msg.sender] = name;

  // Set the last caller name
  lastCallerName = name;
}
```


TCS Confidential

THANK YOU

TATA CONSULTANCY SERVICES