Smart Contract Designing Using Solidity

By

Dr. Aparna Kumari

Aparna.kumari@nirmauni.ac.in

Smart Contract

- A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract.
- Smart contracts allow the performance of credible transactions without third parties.
- These transactions are trackable and irreversible.
- The concept of smart contracts was first proposed by Nick Szabo in 1994. Szabo is a legal scholar and cryptographer known for laying the groundwork for digital currency.

A “dumb contract” example

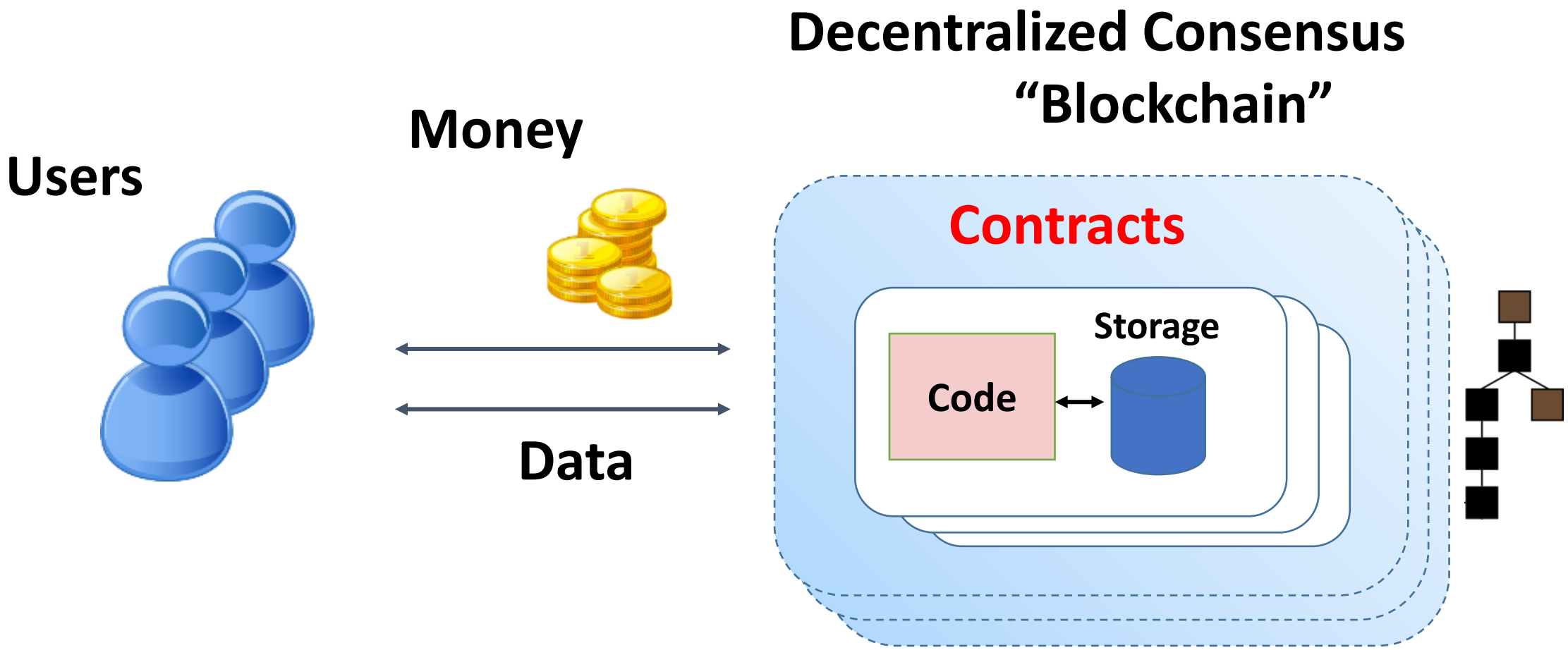
Alice will reveal to Bob a value x such that SHA-256(x) = 0x2a...

In exchange, Bob will give Alice $10 in cash.

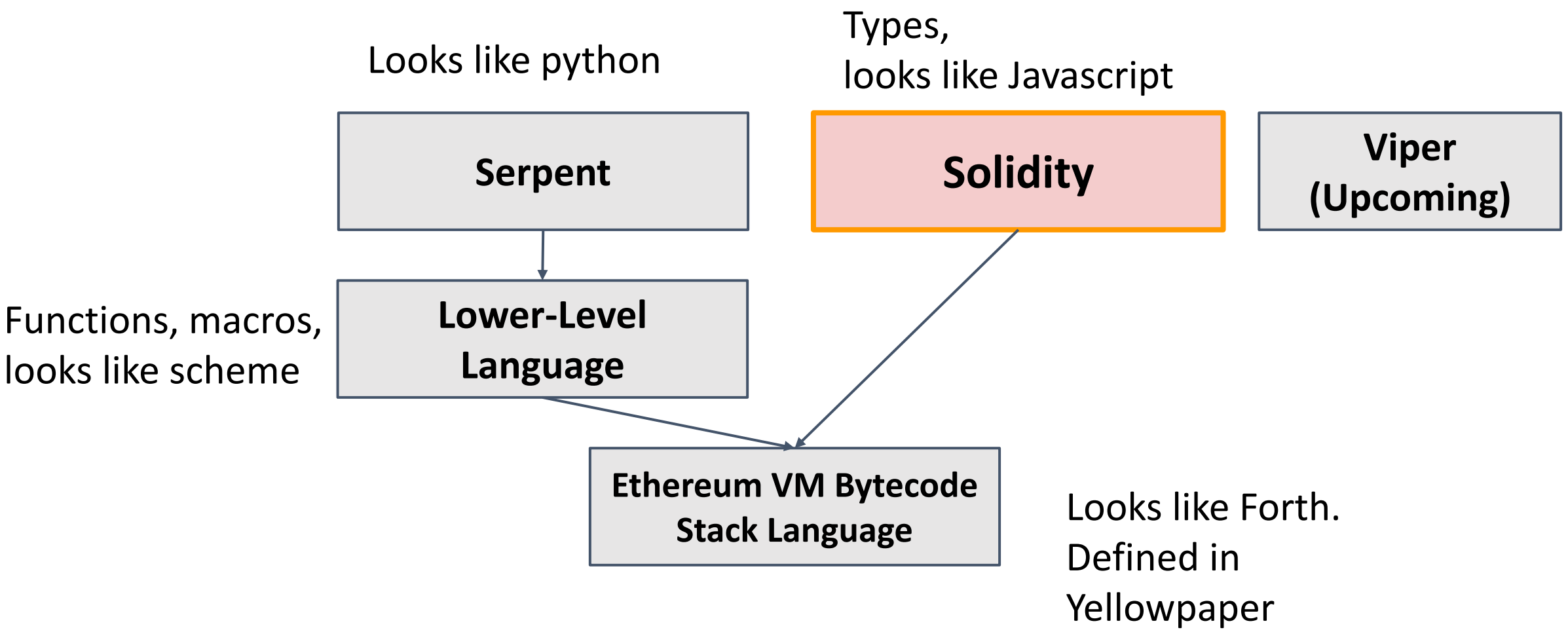
If Alice does not give Bob by July 1, 2018, then she will pay a penalty of US$1 per day that she is late, up to US$100.

Signed: Alice Bob

Smart Contracts: user-defined programs running on top of a blockchain



Ethereum Languages



Solidity

- Solidity is a contract-oriented, high-level programming language for implementing smart contracts.
- Solidity is highly influenced by C++, Python and JavaScript.
- It has been designed to target the Ethereum Virtual Machine (EVM).
- Solidity is statically typed, supports inheritance, libraries and complex user-defined types programming language.
- It helps to create contracts for uses such as voting, crowdfunding, blind auctions, and multi-signature wallets.

Prerequisites

- Familiarity with blockchain, and general programming concept.

Solidity - Environment Setup

- Method 1 - npm / Node.js
  - Install Node.js → Install Solidity compile
- Method 2 - Docker Image
- Method 3: Binary Packages Installation
  - https://soliditylang.org/ → go to Install tab
- Method 4: Online Compiler
  - Remix IDE to Compile and Run our Solidity Code base.

Remix IDE

Remix is the web-based environment for the development and testing of contracts using Solidity.

It is a feature-rich IDE which does not run on live blockchain.

In fact, it is a simulated environment in which contracts can be deployed, tested, and debugged.

It is available at ***https://remix.ethereum.org***.

First Program

- // SPDX-License-Identifier: GPL-3.0
- pragma solidity >=0.4.0 <0.6.0;
- contract SimpleContract {
- uint storedData;
- function set(uint x) public {
- storedData = x; }
- function get() public view returns (uint) {
- return storedData;
- }}

Not compile earlier than version 0.4.0 and it will also not work on a compiler starting from version 0.6.0

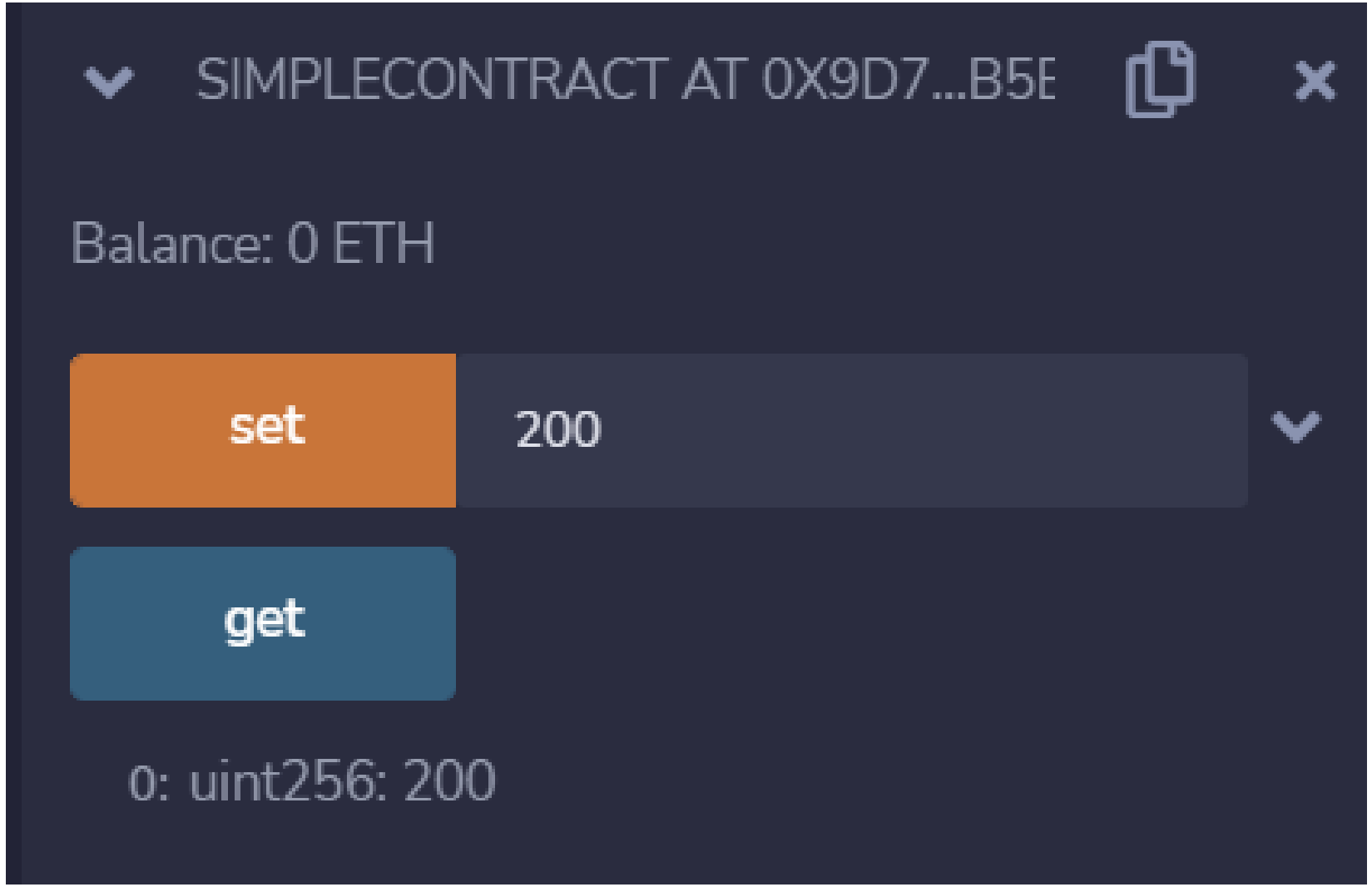
Collection of code (its functions) and data (its state) that resides at a specific address

The line uint storedData declares a state variable called storedData of type uint and the functions set and get can be used to modify or retrieve the value of the variable.

Steps to deploy the Smart Contract

- Go to Remix IDE to Compile and Run our Solidity Code.
- **Step 1** – Copy the given code in Remix IDE Code Section.
- **Step 2** – Under Compile Tab, click **Start to Compile** button.
- **Step 3** – Under Run Tab, click **Deploy** button.
- **Step 4** – Under Run Tab, Select **SimpleContract at 0x...** in drop-down.
- **Step 5** – Click **set** Button after giving input value and the click on the **get** Button to display the result.

Output


SIMPLECONTRACT AT 0X9D7...B5E
Balance: 0 ETH
set
200
get
0: uint256: 200

Output

```
[vm] from: 0x5B3...eddC4 to: SimpleContract.set(uint256) 0x9D7...b5E99 value: 0 wei data: 0x60f...000c8
logs: 0 hash: 0x591...a5f4d                                                                    Debug
call to SimpleContract.get

CALL  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: SimpleContract.get() data: 0x6d4...ce63c   Debug

from                    0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to                      SimpleContract.get() 0x9D7f74d0C41E726EC95884E0e97Fa6129e3b5E99

execution cost          23377 gas (Cost only applies when called by a contract)

input                   0x6d4...ce63c

decoded input           {}

decoded output          {
                            "0": "uint256: 200"
                        }
```