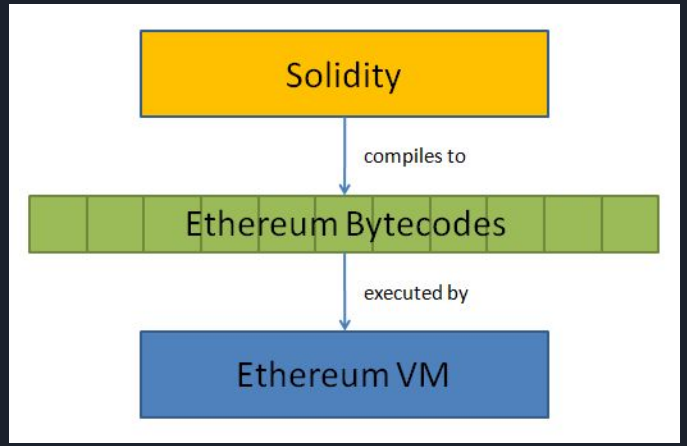
Ethereum Smart contracts

La Blockchain da vicino

Solidity

Il linguaggio Solidity, la cui sintassi si ispira a JS, viene impiegato per la scrittura degli smart-contracts, che potremmo paragonare a classi

Tale linguaggio viene compilato in un linguaggio di livello inferiore, quello esposto dalla Ethereum Virtual Machine. I comandi ricordano molto l'Assembly…PUSH1 0 CALLDATALOAD SLOAD NOT PUSH1 9 JUMPI STOP JUMPDEST PUSH1 32 CALLDATALOAD PUSH1 0 CALLDATALOAD SSTORE



Ethereum Virtual Machine

EVM esegue codice compilato e “caricato” sulla blockchain.

L’esecuzione del codice, che una volta pubblicato non appartiene più a nessuno, corrisponde all’invio di una transazione verso di esso e alla sua attivazione

```
0x00 STOP Halts execution
0x01 ADD Addition operation
0x02 MUL Multiplication operation
0x03 SUB Subtraction operation
0x04 DIV Integer division operation
0x07 SMOD Signed modulo
```

guida su https://ethereum.gitbooks.io/frontier-guide/content/developers.html

Geth

Download Geth:

https://geth.ethereum.org/downloads/

Download Mist:

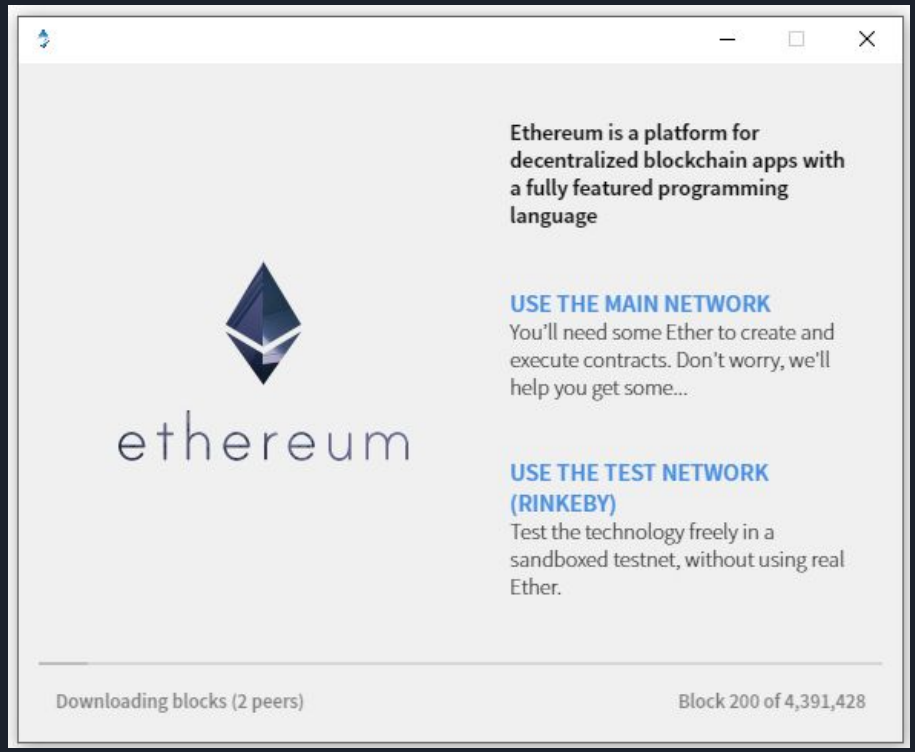
https://github.com/ethereum/mist/releases/tag/v0.9.2

Potete scegliere di compilare i sorgenti oppure utilizzare uno dei binari messi a disposizione. Quindi eseguite geth specificando il file per la socket, la directory per il database e le keystore, il developer mode e che volete una console JS (bleurg)

```
geth --ipcpath geth.ipc --datadir E:\ethTest --dev console
```

Mist

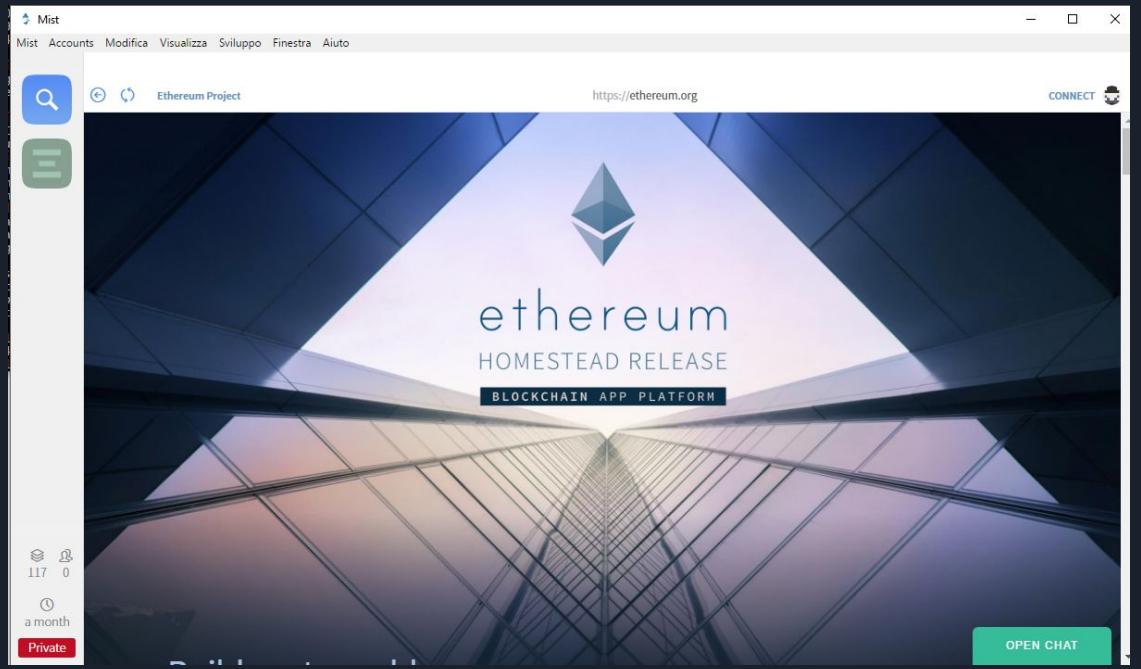
Mist e` la parte visibile della blockchain, un'interfaccia che si appoggia sul run-time geth.



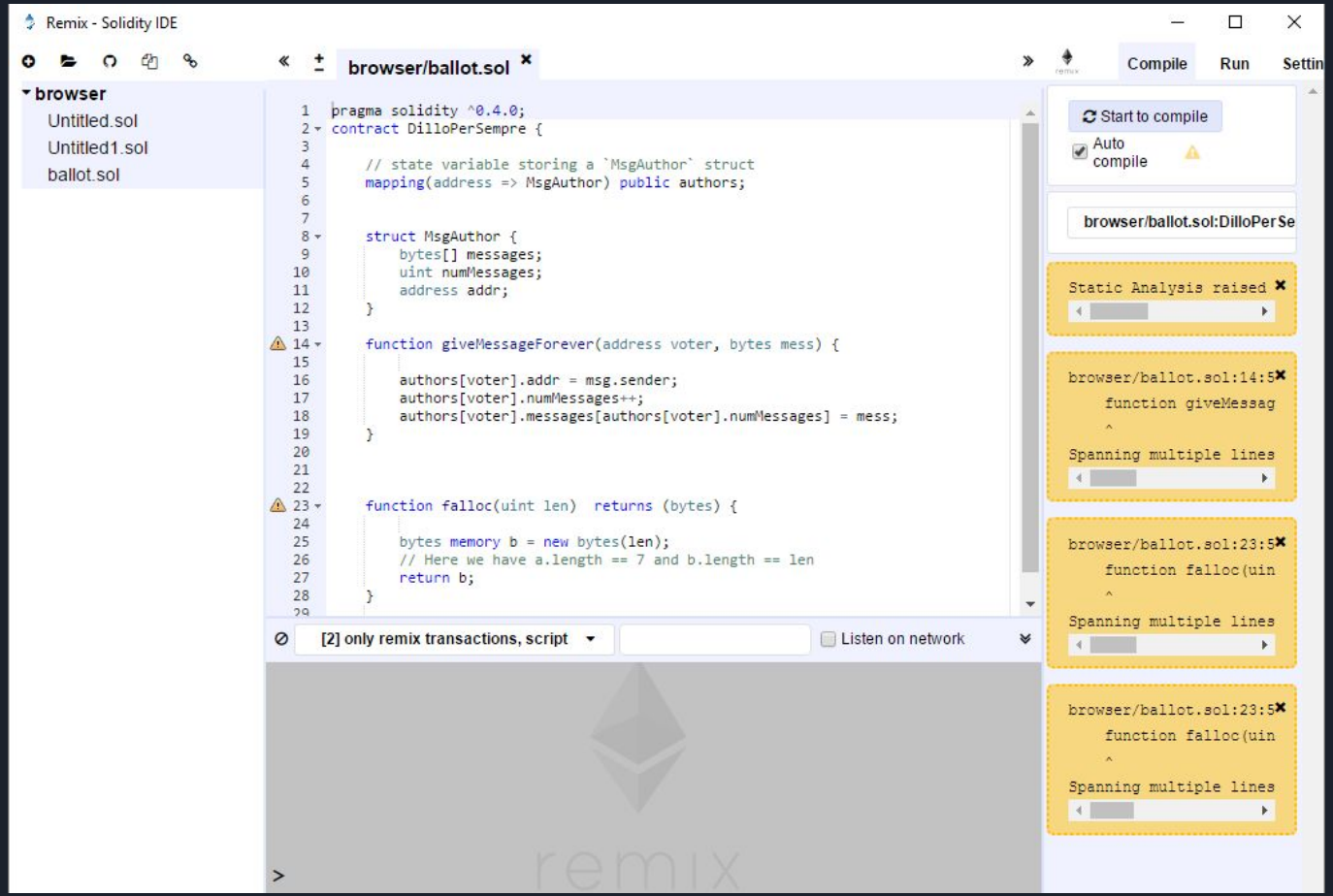
Mist

E` importante iniziare con una blockchain privata

E’possibile creare account (wallet, in soldoni) e riempirli minando: avendo una rete vuota, raggiungere somme rilevanti sara` un istante. In test.



Remix


Remix - Solidity IDE
browser
Untitled.sol
Untitled1.sol
ballot.sol
browser/ballot.sol
Compile
Run
Settin
pragma solidity ^0.4.0;
contract DilloPerSempre {
// state variable storing a `MsgAuthor` struct
mapping(address => MsgAuthor) public authors;
struct MsgAuthor {
bytes[] messages;
uint numMessages;
address addr;
}
function giveMessageForever(address voter, bytes mess) {
authors[voter].addr = msg.sender;
authors[voter].numMessages++;
authors[voter].messages[authors[voter].numMessages] = mess;
}
function falloc(uint len) returns (bytes) {
bytes memory b = new bytes(len);
// Here we have a.length == 7 and b.length == len
return b;
}
Start to compile
Auto compile
browser/ballot.sol:DilloPerSe
Static Analysis raised
browser/ballot.sol:14:5
function giveMessag
Spanning multiple lines
browser/ballot.sol:23:5
function falloc(uin
Spanning multiple lines
browser/ballot.sol:23:5
function falloc(uin
Spanning multiple lines
[2] only remix transactions, script
Listen on network
remix

Inheritance

```
contract mortal {
    /* Define variable owner of the type address */
    address owner;

    /* This function is executed at initialization and sets the owner of the contract */
    function mortal() { owner = msg.sender; }

    /* Function to recover the funds on the contract */
    function kill() { if (msg.sender == owner) selfdestruct(owner); }
}

contract greeter is mortal {
    /* Define variable greeting of the type string */
    string greeting;

    /* This runs when the contract is executed */
    function greeter(string _greeting) public {
        greeting = _greeting;
    }

    /* Main function */
    function greet() constant returns (string) {
        return greeting;
    }
}
```

Creazione Contract

Una volta compilato, cliccando “Create” da REmix appare la finestra per confermare la transazione di creazione

Tale transazione andra` minata usando geth:

miner.start()