Blockchain & Αποκεντρωμένες Εφαρμογές

1η Εργαστηριακή Δραστηριότητα

Μαυρίδης Ιωάννης, Φουληράς Παναγιώτης
Μάστορας Θεόδωρος
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

MULTIMEDIA SECURITY NETWORKING

A. Solidity & πηγές για τα smart contracts.

1. Επισκεφτείτε την ιστοσελίδα https://solidity.readthedocs.io/en/v0.5.3/. Εκεί μπορείτε να βρείτε ένα αναλυτικό εγχειρίδιο της Solidity. Μπορείτε να ξεκινήσετε το διάβασμα από τα τμήματα:

- "A Simple Smart Contract" https://solidity.readthedocs.io/en/v0.5.3/introduction-to-smart-contracts.html#simple-smart-contract
- "Solidity by Example" https://solidity.readthedocs.io/en/v0.5.3/solidity-by-example.html
- "Solidity in Depth" https://solidity.readthedocs.io/en/v0.5.3/solidity-in-depth.html

2. Η Solidity είναι μια ECMAScript γλώσσα προγραμματισμού της οποίας το συντακτικό είναι επηρεασμένο από γλώσσες όπως η C ή η Java. Είναι μιας πλήρης γλώσσα με την έννοια πως μπορεί να αναλάβει οποιαδήποτε λειτουργία (σε αντίθεση π.χ. με την αντίστοιχη του BitCoin στην οποία δεν επιτρέπονται επαναλήψεις). Είναι μια αντικειμενοστραφής γλώσσα, δηλαδή υποστηρίζει κληρονομικότητα, έχοντας τη δήλωση "contract" αντί της δήλωσης "class".

Τα στοιχεία της γλώσσας όπως οι τύποι και οι δομές μοιάζουν πολύ με τη JavaScript αλλά αν επισκεφτείτε το τμήμα "Solidity in Depth", στη σελίδα που αναφέρθηκε νωρίτερα στο [1], και στη «διαδρομή»:

`in Depth -> Units -> Special var and functions -> Block & Trans Properties`

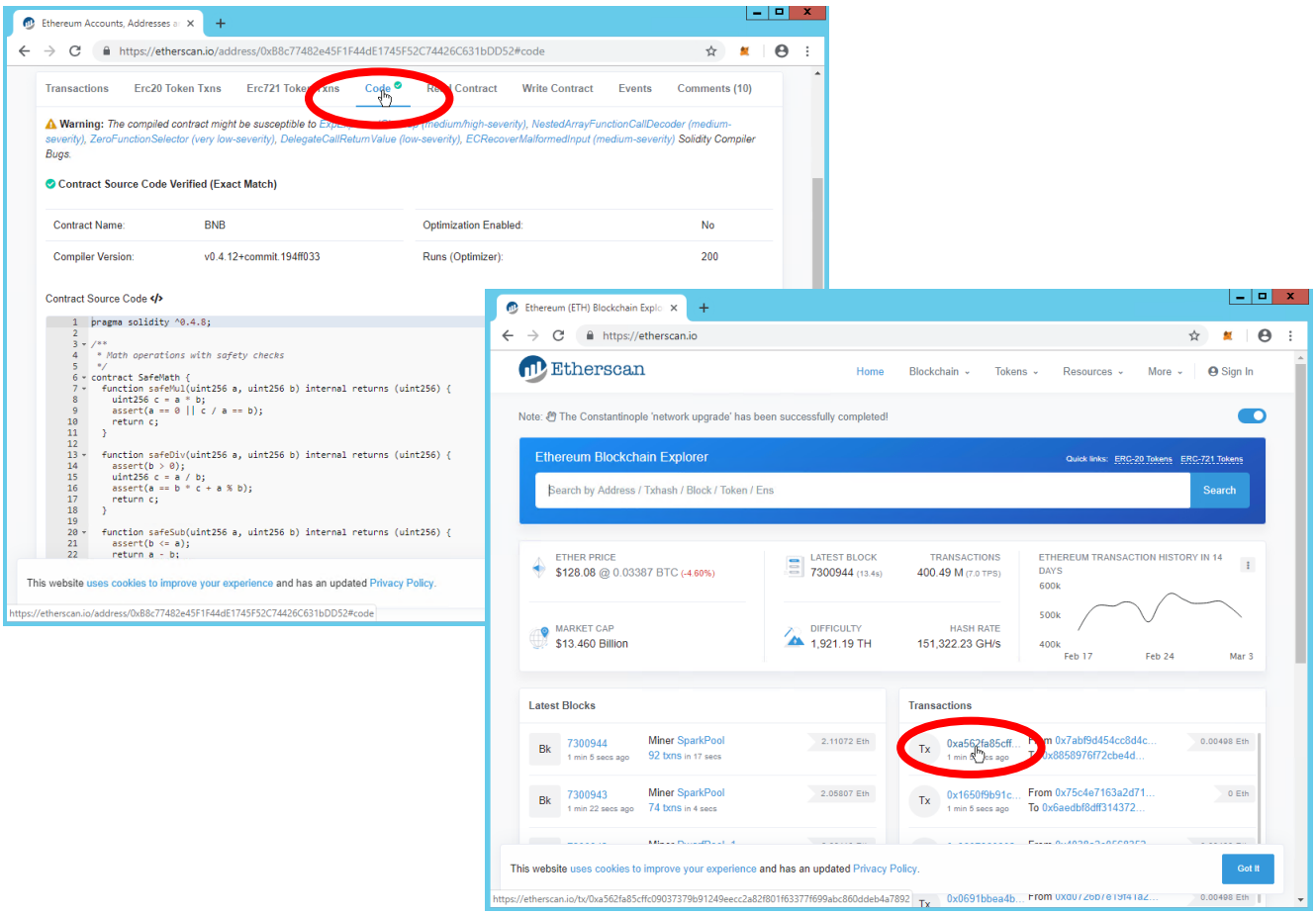
μπορείτε να δείτε μια σειρά από ενσωματωμένα αντικείμενα της γλώσσας που αφορούν την Blockchain και τις λειτουργίες της. Για παράδειγμα η ιδιότητα **`block.coinbase`** είναι η διεύθυνση του miner ο οποίος καταχώρησε το block μέσα στο οποίο φυλάσσεται το smart contract (δηλαδή το πρόγραμμα). Ενώ κάθε φορά που καλείται μια μέθοδος μεταφοράς «χρημάτων», αυτόματα στις παραμέτρους της κλήσης καταχωρείται η διεύθυνση αυτού που την καλεί στην ιδιότητα **`msg.sender`** και του ποσού που «μεταφέρει» στην **`msg.value`**.

Αντίστοιχα στη «διαδρομή»:

…**`in Depth -> Units -> Special var and functions -> Math & Crypto functions`**

μπορείτε να δείτε μια σειρά από ενσωματωμένες συναρτήσεις κρυπτογραφίας και μαθηματικών. Για παράδειγμα η συνάρτηση **`keccak256`** αποτελεί την κύρια συνάρτηση hash του Ethereum blockchain.

3. Επισκεφτείτε τη σελίδα https://etherscan.io/tokens. Τα tokens είναι smart contracts που υλοποιούν κάποια μετοχή ή κρυπτονόμισμα. Ακολουθήσετε το Hash κάποιας συναλλαγής με κάποιο token. Ακολουθήστε το Hash του ίδιου του token. Μπορείτε να δείτε τον πηγαίο του κώδικα σε Solidity; Επισκεφτείτε τη σελίδα https://etherscan.io/. Παρατηρήστε κάθε πότε γράφεται ένα νέο block ή πόσες συναλλαγές εκτελούνται το δευτερόλεπτο. Μπορείτε να εντοπίσετε συναλλαγές με smart contracts;



4. Κάθε κόμβος της Blockchain εκτελεί το δικό του αντίγραφο της εικονικής μηχανής EVM στην οποία εκτελούνται τα smart contracts. Τα smart contracts:

- αποθηκεύονται σε εκτελέσιμη μορφή bytecode στην Blockchain.
- συνήθως είναι γραμμένα σε Solidity αλλά υπάρχουν και άλλες γλώσσες λιγότερο δημοφιλείς (Lisk, Bamboo, Viper)
- δεν υποστηρίζουν «μόνιμη» μνήμη (π.χ. αρχεία)
- μπορούν να αποθηκεύουν και να διατηρούν μεταβλητές στην Blockchain. Η εκτέλεση εντολών ή η εγγραφή τιμών στις «μεταβλητές» κοστίζει Gas (η νομισματική μονάδα είναι το Wei).

Unit	Wei Value	Wei
wei	1 wei	1
Kwei (babbage)	1e3 wei	1,000
Mwei (lovelace)	1e6 wei	1,000,000
Gwei (shannon)	1e9 wei	1,000,000,000
microether (szabo)	1e12 wei	1,000,000,000,000
milliether (finney)	1e15 wei	1,000,000,000,000,000
ether	1e18 wei	1,000,000,000,000,000,000

Για περισσότερες πληροφορίες για την EVM μπορείτε να επισκεφτείτε τη σελίδα https://github.com/ethereum/wiki/wiki/Ethereum-Virtual-Machine-(EVM)-Awesome-List.

B. Ανάπτυξη smart contract με το remix.

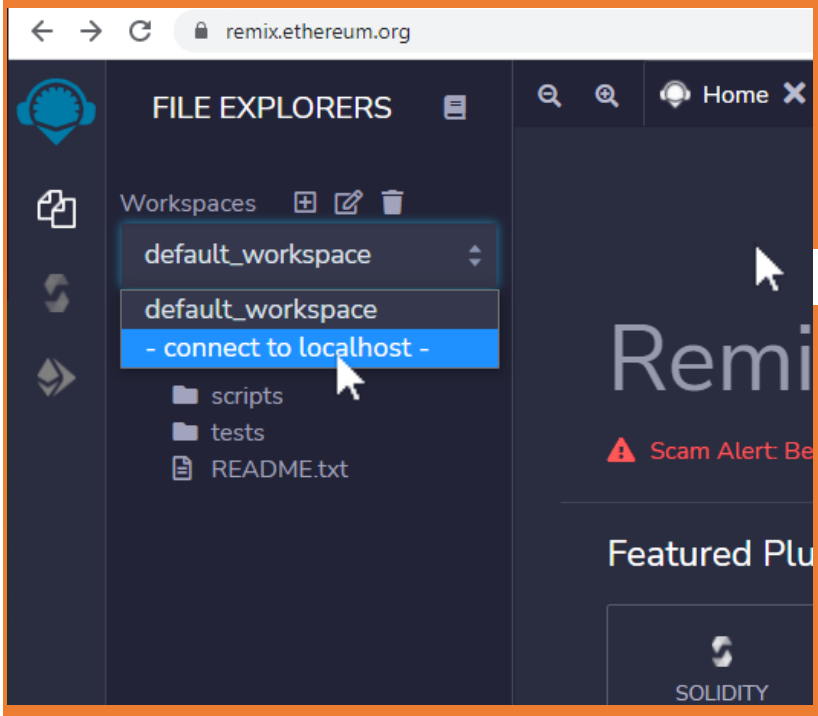
1. Επισκεφτείτε την ιστοσελίδα http://remix.ethereum.org/. Η ιστοσελίδα 'remix' αποτελεί ένα web based IDE για τη γλώσσα Solidity. Εκτός από τη μεταγλώττιση (σε bytecode):

- διευκολύνει την εκτέλεση δοκιμών και ελέγχων στο συμβόλαιο, πριν την τελική δημοσίευσή του
- αναλαμβάνει τη μετανάστευση του συμβολαίου τόσο στην κύρια όσο και σε δοκιμαστικές ή ιδιωτικές blockchain (όπως η Sepolia, το Ganache, ή το Truffle)
- αναγνωρίζει την ύπαρξη του Metamask και δρομολογεί όλες τις απαραίτητες συναλλαγές (όπως τη χρέωση εγγραφής του συμβολαίου) από τον επιλεγμένο λογαριασμό
- προετοιμάζει ένα «περιβάλλον πελάτη» για το συμβόλαιο, δηλαδή, μια διασύνδεση μέσα από την οποία καλούνται συναρτήσεις του συμβολαίου, δίνοντάς τους ορίσματα
- επιτρέπει την πρόσβαση και εγγραφή στη swarm ή την ipfs, δηλαδή την «εξωτερική - περιφερειακή» μνήμη της Ethereum VM όπου συνήθως αποθηκεύονται μόνιμα οι εφαρμογές διασύνδεσης των συμβολαίων (οι DApps)

npm install -g @remix-project/remixd

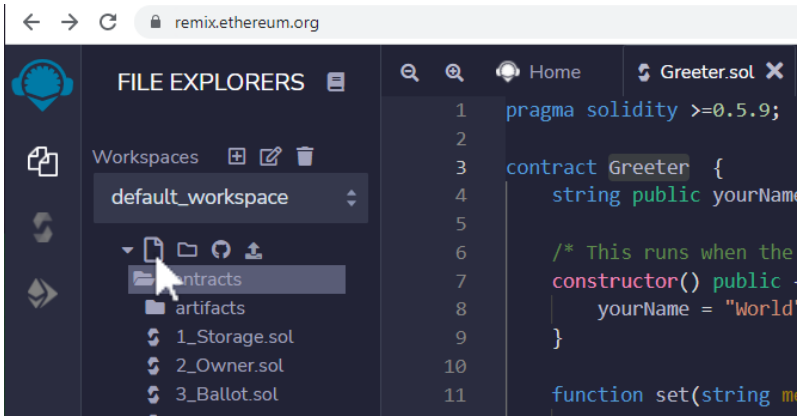
remixd -s <absolute-path-to-the-shared-folder> --remix-ide https://remix.ethereum.org

```
[INFO] you are using the latest version 0.5.6
[WARN] You may now only use IDE at https://remix.ethereum.org to connect to that instance
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE

[INFO] Sat Feb 19 2022 20:14:10 GMT+0200 (GMT+02:00) remixd is listening on 127.0.0.1:65520
[INFO] Sat Feb 19 2022 20:14:10 GMT+0200 (GMT+02:00) slither is listening on 127.0.0.1:65523
```



Προσοχή, το remix αλλάζει συνεχώς καθώς προστίθενται νέες δυνατότητες πολύ συχνά.

2. Με επιλεγμένο το φάκελο contracts, κάντε κλικ στο εικονίδιο της «σελίδας» και δημιουργήστε ένα νέο αρχείο με το όνομα “Greeter.sol”.

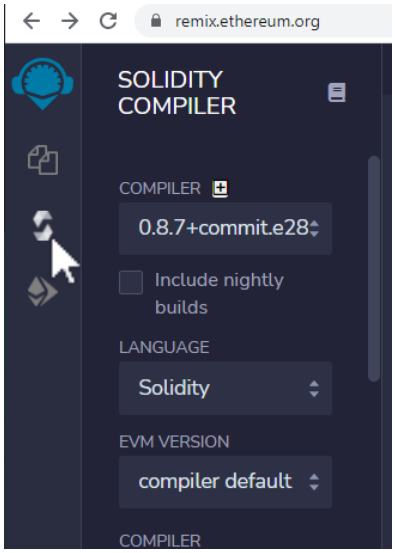


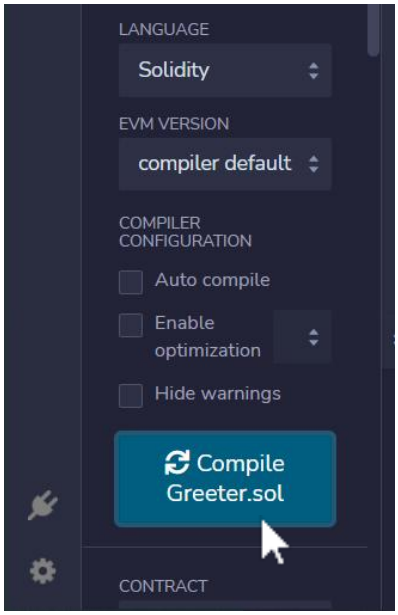
Πληκτρολογήστε ή αντιγράψτε το παρακάτω πρόγραμμα Solidity: link

```
pragma solidity >=0.5.9;

contract Greeter  {
    string public yourName;  // data

    /* This runs when the contract is executed */
    constructor() public {
        yourName = "World";
    }

    function set(string memory name) public {
        yourName = name;
    }

    function hello() view public returns (string memory) {
        return yourName;
    }
}
```

Παρατηρήστε ως κλάση το συμβόλαιο. Έχει μία public μεταβλητή την “yourName” και δύο public μεθόδους, την “set” και την “hello”. Σκεφτείτε τη συμπεριφορά του.

3. Στο αριστερό τμήμα του remix IDE, επιλέξτε την καρτέλα “Compiler”. Πατήστε το κουμπί και κάνετε μεταγλώττιση του συμβολαίου.



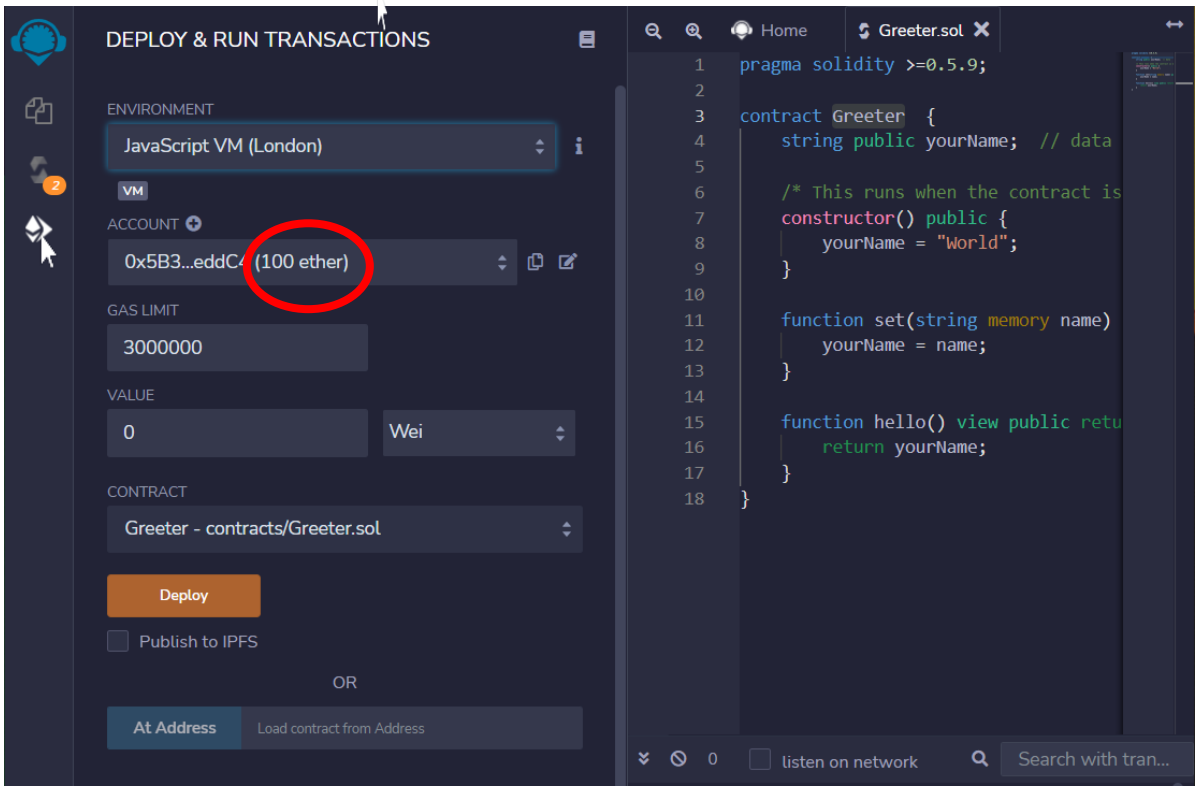


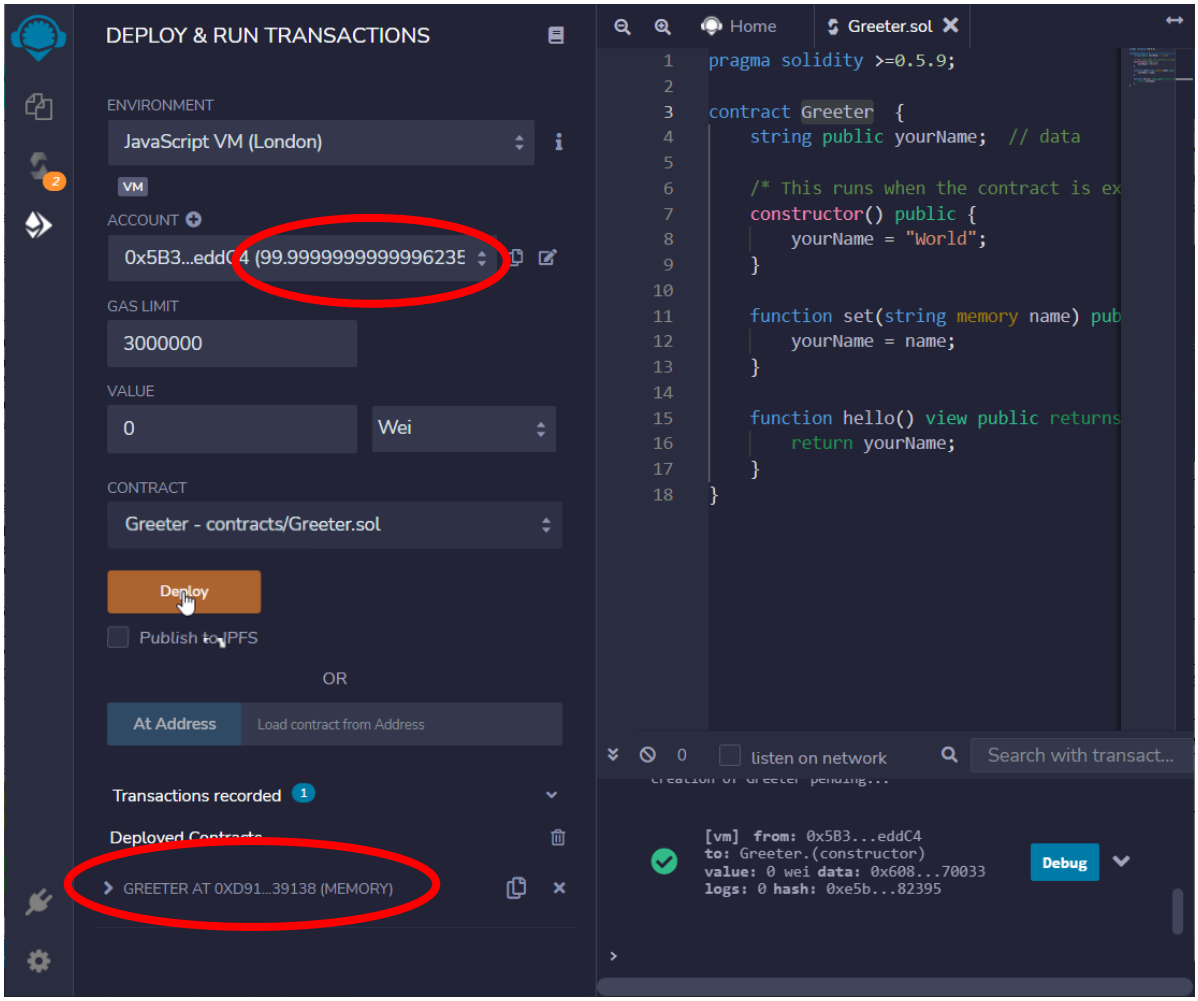
Παρατηρήστε τα παραγόμενα αρχεία πατώντας το κουμπί “Compilation Details”.

4. Στο αριστερό τμήμα, επιλέξτε την καρτέλα “**Deploy & Run**”. Από την πτυσσόμενη λίστα “Environment”, επιλέξτε “**Remix VM (Cancun)**”. Το remix με αυτό τον τρόπο συνδέεται με μια ιδιωτική τοπική blockchain που υλοποιεί μέσα στον ίδιο τον browser σας. Για τις ανάγκες επικοινωνίας με αυτήν την blockchain, σας δίνει και 10 πορτοφόλια με 100 ether το καθένα. Η blockchain αυτή κάνει auto mine ένα μπλοκ, κάθε φορά που η κατάστασή του αλλάζει. Η ταχύτητα εκτέλεσης του συμβολαίου σας (των κλήσεων προς τις μεθόδους του) είναι πολύ πιο γρήγορη σε σχέση με την περίπτωση που αυτό δημοσιευτεί σε μια πραγματική blockchain όπου κάθε μπλοκ επικυρώνεται κάθε 15 δευτερόλεπτα.

Αν στην πτυσσόμενη λίστα “Environment”, επιλέξετε “**Injected Web3**”, το remix μπορεί να συνδεθεί με την blockchain που είναι επιλεγμένη στο Metamask. Αν επιλέξετε “**Web3 Provider**”, μπορεί να συνδεθεί με την blockchain σε κάποια συγκεκριμένη διεύθυνση ip (όπως η τοπική σας διεύθυνση, αν π.χ. εκτελείτε στον υπολογιστή σας το truffle ή το Ganashe). Στα παραδείγματα στη συνέχεια θα χρησιμοποιούμε την “ **Remix VM (Cancun)**”

Για να «δημοσιεύσετε», δηλαδή να «εγκαταστήσετε» το συμβόλαιό σας στην συνδεμένη blockchain, πρέπει να πληρώσετε. Πατώντας το κουμπί “Deploy”, θα δείτε στην ετικέτα “Deployed Contracts” να εμφανίζεται η διεύθυνση του συμβολαίου σας. Θα παρατηρήσετε επίσης να μειώνεται ελάχιστα το απόθεμα του επιλεγμένου πορτοφολιού.


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM (London)
VM
ACCOUNT
0x5B3...eddC4 (100 ether)
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
Greeter - contracts/Greeter.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Home
Greeter.sol
pragma solidity >=0.5.9;
contract Greeter {
string public yourName; // data
/* This runs when the contract is
constructor() public {
yourName = "World";
}
function set(string memory name)
yourName = name;
}
function hello() view public retu
return yourName;
}
}
0
listen on network
Search with tran...


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM (London)
VM
ACCOUNT
0x5B3...eddC4 (99.999999999999623E
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
Greeter - contracts/Greeter.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 1
Deployed Contracts
GREETER AT 0XD91...39138 (MEMORY)
Home
Greeter.sol
pragma solidity >=0.5.9;
contract Greeter {
string public yourName; // data
/* This runs when the contract is ex
constructor() public {
yourName = "World";
}
function set(string memory name) pub
yourName = name;
}
function hello() view public returns
return yourName;
}
}
0
listen on network
Search with transact...
[vm] from: 0x5B3...eddC4
to: Greeter.(constructor)
value: 0 wei data: 0x608...70033
logs: 0 hash: 0xe5b...82395
Debug

Στο δεξιό κάτω τμήμα, κάτω από τον editor, εμφανίζεται η διεύθυνση της συναλλαγής για την εγκατάσταση του συμβολαίου. Σπουδαιότερη όμως είναι η διεύθυνση του ίδιου του συμβολαίου που εμφανίζεται αριστερά ως πτυσσόμενη λίστα. Σημειώστε την προσωρινά σε κάποιο αρχείο κειμένου.

5. Αναπτύξτε την πτυσσόμενη λίστα με τη διεύθυνση του συμβολαίου, κάτω από την ετικέτα “Deployed Contracts”. Μπορείτε από εκεί να εκτελέσετε το συμβόλαιό σας καλώντας τις συναρτήσεις του (τις μεθόδους του). Το remix έχει ετοιμάσει μια απλή διασύνδεση με ένα κουμπί για κάθε συνάρτηση. Την κλήση των συναρτήσεων θεωρείται πως την κάνει ο λογαριασμός που έχει επιλεγεί στο πεδίο “Account” (ψηλά στο ίδιο παράθυρο). Από αυτόν τον λογαριασμό αφαιρούνται χρήματα για την εκτέλεση κλήσεων, όταν αυτές τροποποιούν την κατάσταση του συμβολαίου.

Παρατηρήστε πως για κάθε public μεταβλητή (εδώ την “yourName”), η solidity χτίζει μια μέθοδο getter. Η “hello” που διάβαζε τη μεταβλητή ήταν περιττή.

Πειραματιστείτε με κλήσεις στο συμβόλαιο.

C. Δεύτερο παράδειγμα smart contract.

1. Επιστρέψτε στον "File explorer" και δημιουργήστε ένα νέο αρχείο solidity με το όνομα "SimpleStorage.sol".

Πληκτρολογήστε ή αντιγράψτε το παρακάτω πρόγραμμα Solidity: **link**

```
pragma solidity >=0.5.9;
// Imagine a big integer that the whole world could share
contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() view public returns (uint) {
        return storedData;
    }

    function increment (uint n) public {
        storedData = storedData + n;
        return;
    }

    function decrement (uint n) public {
        storedData = storedData - n;
        return;
    }

}
```

2. Σκεφτείτε τη συμπεριφορά του (προσπαθήστε να την καταλάβετε). Παρατηρείστε ότι δεν υπάρχει ρητός constructor, ωστόσο πάντα δημιουργείται αυτόματα ένας από προεπιλογή. Κάντε μεταγλώττιση, δημοσιεύστε το στην τοπική " Remix VM (Cancun)" και τέλος εκτελέστε το από το παραγόμενο interface. Στο παραγόμενο interface εμφανίζονται όλες οι public μεταβλητές και μέθοδοι. Σημειώστε κάπου τη διεύθυνση όπου δημοσιεύτηκε και αυτό το συμβόλαιο.

3. Χρησιμοποιήστε τη διεύθυνση του Greeter που είχατε σημειώσει. Εντοπίστε το και εκτελέστε το και πάλι. Χρησιμοποιείστε τη διεύθυνση του SimpleStorage για να επιστρέψετε πάλι σε αυτό. Πειραματιστείτε με το SimpleStorage.

4. Από την πτυσσόμενη λίστα "Environment", επιλέξτε "**Injected Web3**" και με τη βοήθεια του Metamask δημοσιεύστε το SimpleStorage στο δοκιμαστικό δίκτυο Goerli. Με χρήση της διεύθυνσής του, εντοπίστε το στην Etherscan https://goerli.etherscan.io/. Συγκρίνεται το ABI και το Bytecode με αυτά του remix. Κάντε κάποιες καταχωρίσεις/εκτελέσεις και παρατηρήστε τους χρόνους και τις χρεώσεις στο πορτοφόλι σας σε σχέση με πριν.

5. Από την πτυσσόμενη λίστα "Environment", επιλέξτε **Web3 Provider**" και δημοσιεύστε το SimpleStorage στο τοπικό Ganache. Στο Ganache γίνεται επίσης auto mine και η εκτέλεση κλήσεων στο συμβόλαιο είναι ταχύτατη. Σημειώστε τη διεύθυνση δημοσίευσης του συμβολαίου στο

Ganache. Συνδέστε το Metamask με το τοπικό Ganache (θα χρειαστεί import κάποιου από τα πορτοφόλια του Ganache) και δοκιμάστε να εντοπίσετε το συμβόλαιο και να το εκτελέσετε στην ίδια διεύθυνση (έχοντας πλέον "**Injected Web3**" στο "Environment").