­­­How to setup your Ethereum Development Environment.

For Windows 10 and Ubuntu 16.04+

[INTRODUCTION to be added]

Ethereum blockchain provides already a wide range of open source products that are meant to help the developers in their journey to discover and build the decentralized world. In this article we will describe end to end a simple way for the developers to connect and communicate with the Ethereum blockchain by using Infura, Metamask, Truffle and NodeJS.

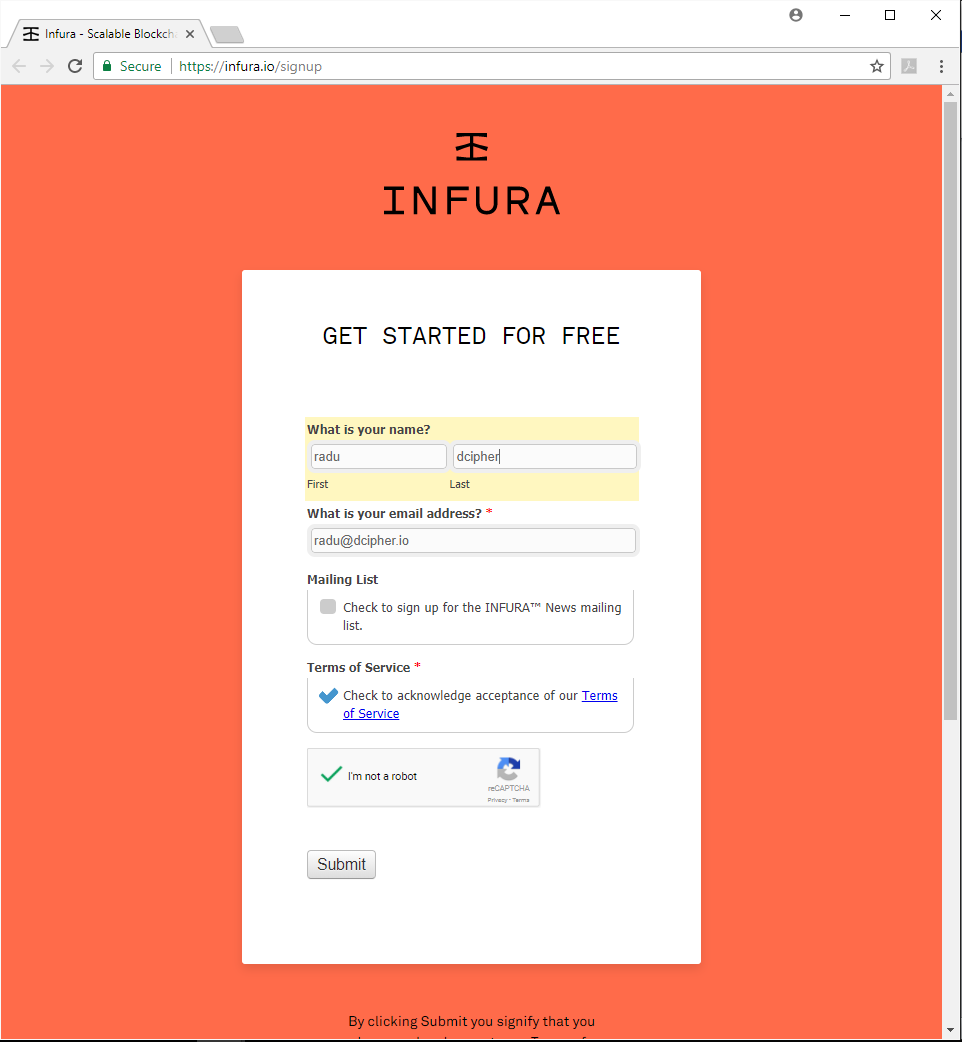
1. Registering with Infura.

We will start with the Infura environment, which, for those of you that didn`t heard of it, they describe themselves as:

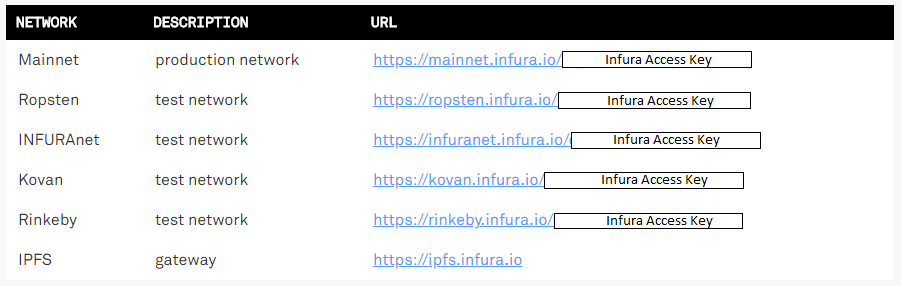
“***We provide secure, reliable, and scalable access to Ethereum APIs and IPFS gateways.”***

*“Infura was born with the goal of delivering stable and reliable RPC access to the internal projects within ConsenSys. As the Ethereum network began to grab the majority of blockchain developer mindshare, we recognized that Infura would be of broad interest to the entire ecosystem.”*

Basically Infura it is a set of API`s to connect to an Ethereum node, without running one locally, so without downloading the whole blockchain, which, if we speak about the main net, is already > 50 GB (but it depends on the way in which you run the node, for instance if you run a parity light node it will be around 15 GB).

We will start by registering a new free account on [Infura website](https://infura.io/signup).

After the successful registration on Infura website you will be redirected to a page containing the links to access various Infura hosted blockchain Nodes.



You will receive the same links in the welcome email Infura will send you after registration.

For our exercise we will use this account to deploy smartcontracts on Ropsten TestNet, and this is a recommendation also for all developers, don`t try to develop directly in the main net, start by doing it on one of the testnet versions as you would like to avoid any risk to lose money before finishing the development and auditing the code properly.

https://ropsten.infura.io/[your\_infura\_access\_key]

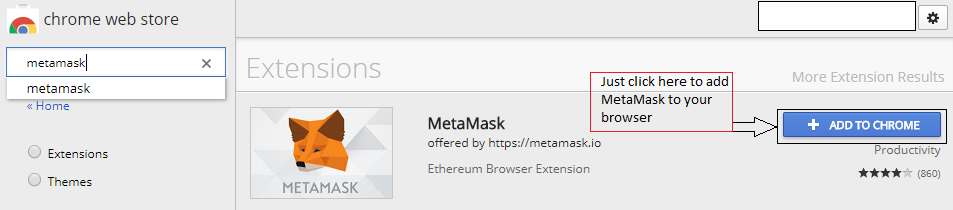
1. Setup Metamask, create your wallet and get funds.

In order to manage to deploy any Smart Contracts on any Ethereum node, Ropsten included, we will need an Ethereum Wallet.

MetaMask is one of the best options for creating and Ethereum account and storing and transferring Ethers and it is a browser extension for Chrome and Firefox web browsers, so easy to use and install as you will see below.

### Installing MetaMask.

To install Metamask simply go to the [Chrome Web Store](https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn) or [Firefox Addons](https://addons.mozilla.org/en-US/firefox/addon/ether-metamask/?src=search) and click on “ADD TO CHROME/FIREFOX”.



### Creating a new wallet.

To create a new wallet in MetaMask simply click on the MetaMask icon in the right upper corner of your browser



You will need to accept the “Terms and Conditions” , insert a new password, confirm the password and save those 12 mnemonic words. In this way, by using those words, you can recover your wallet access in case something happens with it (lost device, forgot password, etc.)

Once you completed the steps above MetaMask should be installed and the wallet created, which now will be empty. In order to deploy a smart contract we will need to get some funds.

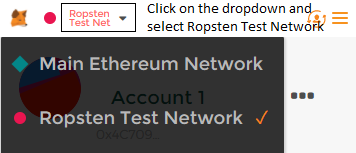
### Getting Funds.

If we want to deploy Smart Contracts or to run transactions on Ethereum we will need to pay fees with Gas.

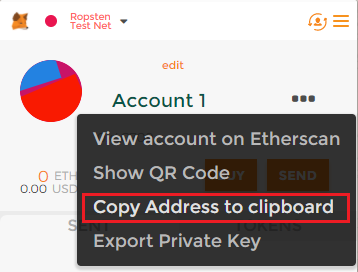
Note: In order to register a transaction or to deploy a smart contract to the network you will need to pay an execution fee in Gas which has a variable conversion rate to ETH over time, depending on the network usage in that specific moment.

The easiest way to get test ether is to use a faucet service. You can find many faucet websites on Google but for now we will use “[Ethereum Ropsten Faucet](http://faucet.ropsten.be:3001/)”

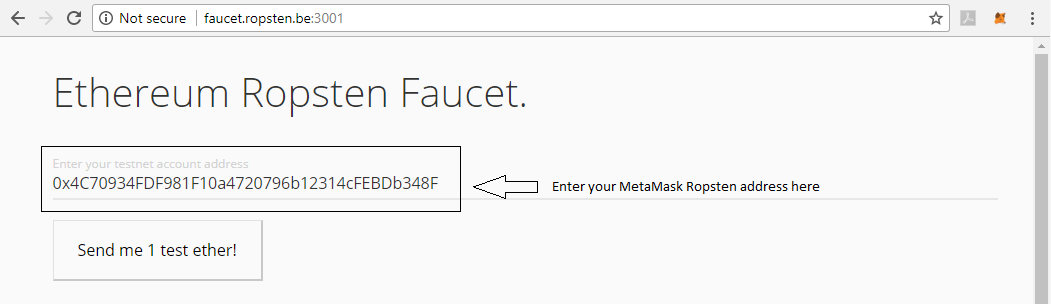
To get your Ropsten Wallet address click on MetaMask icon in your browser , select Ropsten Test Net



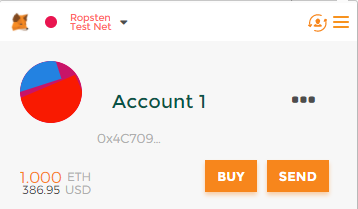
After you selected Ropsten, click in the dotted menu  and select **Copy Address to clipboard.**



Now we can use our address to insert it into the faucet site and click the Send button to receive TestEther.



If everything went well, a few minutes later, our wallet balance should be 1 ETH ☺.



Now that we are done in setting our Infura account and MetaMask add-on, let’s set up the development environment.

1. Installing NodeJS and NPM

First we will need to install **NodeJS** and **NPM.**

For windows users:

1. Download installer package from <https://nodejs.org/en/>

2. Run the installer(the .msi file you just downloaded) and follow the prompts in the installer.

Note: The recommended version at the time this tutorial was written is 8.11.1 so we will use this one.

For Ubuntu users:

**Node.js** is available from the NodeSource’s Debian and Ubuntu binary distributions repository.

1. To add the repository open a Terminal and run the command:

curl -sL https://deb.nodesource.com/setup\_8.x | sudo -E bash -

1. After adding the repository we will run the next command to install NodeJS 8 on our system:

sudo apt-get install -y nodejs

Note: The Node.js package contains the Node.js binary as well as NPM, so you don't need to install NPM separately. However, in order for some NPM packages to work, you will need to install the **build-essentials** package.

1. Git Installation

Note:  According to their website, Git is a distributed version control system, meaning your local copy of code is a complete version control repository.

For windows users:

To install Git on Windows you need to download the latest Git installer from the official website:

<https://gitforwindows.org/>

After the download is completed, run the .exe installer and follow the Setup Wizard to complete the installation.

For Ubuntu users:

To install Git on Ubuntu we will use the APT package manager and run the following commands:

**$ sudo apt-get update**

**$ sudo apt-get install git**

1. Install and configure Truffle

Truffle is a development environment, build by Consensys with the purpose to make create an easy way for the developers to interact, develop and deploy smart contracts in the Ethereum network.

1. To install truffle globally, open a terminal window and run the following command(both for windows and unix):

**$ npm install -g truffle**

*Note: On Windows, if you get the “MS BUILD” error, open and administrator CMD window and install Windows Build Tools with this command:* ***npm install -g windows-build-tools***

1. After the installation is done , in the same terminal, we will set the path for the build tools running this command:

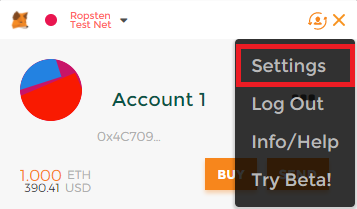
**set PATH="C:\Program Files (x86)\MSBuild\14.0\Bin"**

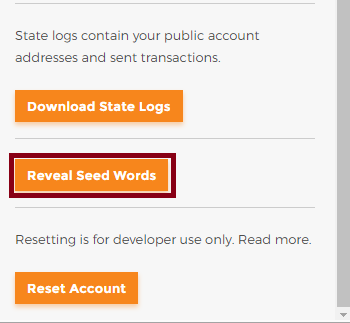
1. In order to use Infura’s Ropsten TestNet , in your project folder we need to install Truffle HDwallet Provider.

Note: “HD Wallet-enabled Web3 provider. Use it to sign transactions for addresses derived from a 12-word mnemonic.”

**$ npm install truffle-hdwallet-provider**

1. The next step is to modify your project’s truffle.js to add a condition to check for the installation of the HDWallet Provider and add your 12-word mnemonic of the account you want to use to deploy on Ropsten (in order to use it to pay the gas fees from):

NOTE: To find your 12-word mnemonic, go in MetaMask, click on the Menu button, click on Settings, click on “Reveal Seed Words”, enter your password and copy those 12 words.



So, your project’s truffle.js file should look something like this, the only thing you need to modify is the value of the mnemonic variable and set it with your own 12 words from MetaMask:

var HDWalletProvider = require("truffle-hdwallet-provider");

var mnemonic = "tonight column vote soap random engaged lion set spirit master execute dad";

module.exports = {

networks: {

development: {

host: "localhost",

port: 8545,

network\_id: "\*" // Match any network id

},

ropsten: {

provider: new HDWalletProvider(mnemonic, "https://ropsten.infura.io/"),

network\_id: 3,

gas: 471238

}

}

};

1. Using Truffle on Ropsten

Good, now that everything is installed and (hopefully) ready to go, let’s test the installation.

To test it, we are going to unbox an example set of smart contracts provided by Truffle, modify truffle.js with the correct settings, deploy the smart contracts on Infura’s Ropsten TestNet and interact with them.

The test will be done in Ubuntu 16.04 but the workflow and commands are the same in Windows.

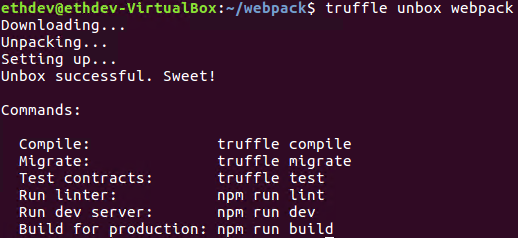
1. First of all let’s create a new directory for our project using **mkdir** command.

**$ mkdir webpack**



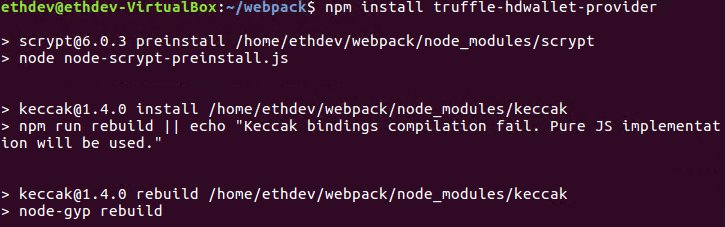
1. When you are inside the newly created directory, lets unbox the sample webpack with the following command:

**$ truffle unbox webpack**



1. Next we must install in the project’s folder the “**truffle-hdwallet-provider**” using the command:

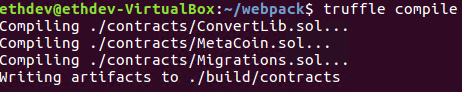
**$ npm install truffle-hdwallet-provider**





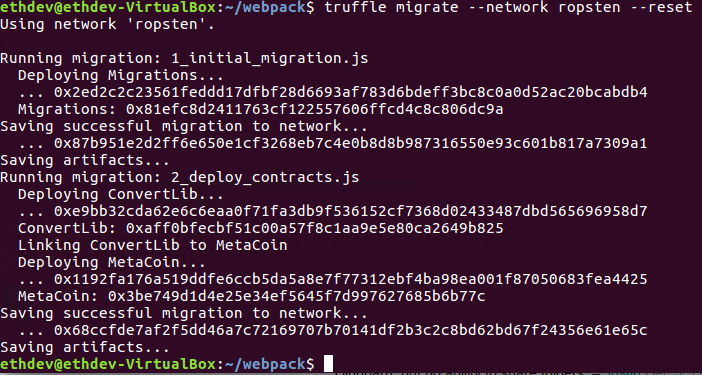
1. Once the installation is done, we must edit truffle.js to use truffle-hdwallet-provider and to import our mnemonic phrase. (please refer to point ”5.d” in this tutorial if you don’t know how to do that).
2. After we are done editing truffle.js we are ready to compile our project with the command:

**$ truffle compile**



1. If everything is ok and the compilation process didn’t thrown any errors we can deploy the contracts to the network with the command:

**$ truffle migrate –network ropsten --reset**



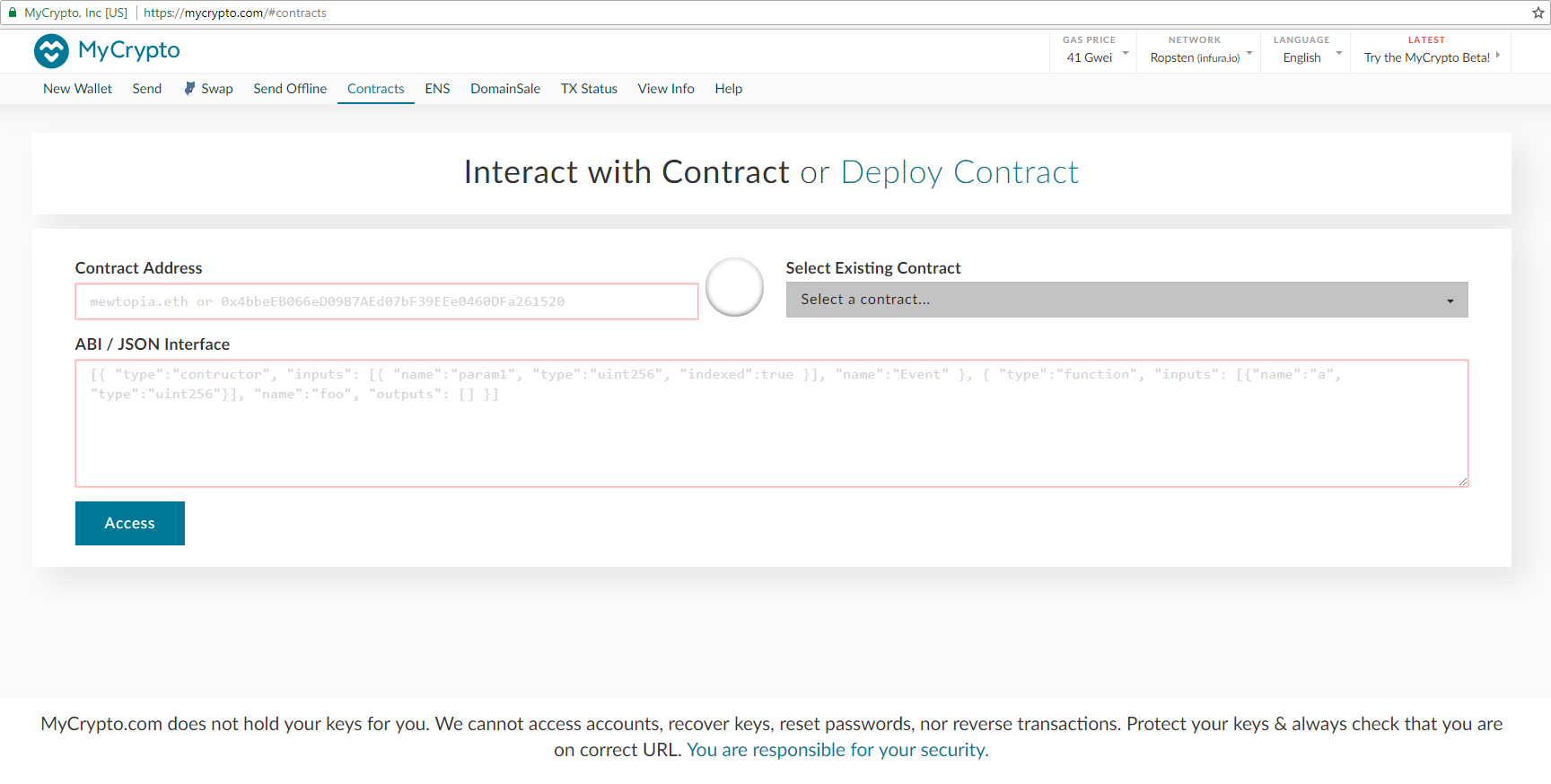
If your message looks the same as the message above…CONGRATS!! You have deployed your first smart contract on the Ethereum TestNet!

7. Interacting with our deployed smart contracts

There are many ways to interact with our deployed smart contracts, starting with the base client Mist, or third party ones parity, MyCrypto, and many others.

For our example we will use the website [MyCrypto.com](https://mycrypto.com/#contracts) .

Note: MyCrypto.com is a forked version of the old MyEtherWallet.com, which is a multi-purpose Ethereum platform.



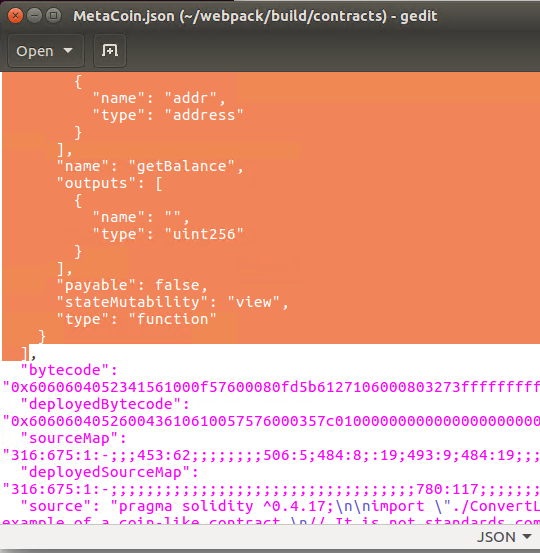
Steps to interact with the contract:

1. Navigate to **MyCrypto.com**
2. Go to **Contracts** section.
3. Change the network to **Ropsten** **(infura.io)**.
4. Insert in the field **“Contract Address”** the address truffle reported upon the creation of the contract(see point 6.f)
5. Insert in the **“ABI / JSON Interface”** field the abi of the contract.

Note: You can find the abi in your project folder at path:

**<Your\_project\_folder>**/build/contracts/**<name\_of\_the\_contract>.**json

The ABI starts after **“abi”:** and ends before **,”bytecode”:** as you can see in the pictures below:



1. After we finished inserting the required field we need to click on “Access” button to access the functions of the contract.

