1. Introduction
   1.1 How to Create Ethereum-based dApps
   1.2 What is a Smart contract?
   1.3 An Overview of dApp Builder and its Smart Contracts

2. Working with dApp Builder
   2.1 Registration with dApp Builder
   2.2 Creating dApps with dApp Builder
   2.3 Working with your dApps
   2.4 Working with your dApps on mobile, mobile dApp widget

3. Supported dApps
   3.1 Voting
   3.2 Escrow
   3.3 Multisignature Wallet
   3.4 Betting
   3.5 Custom Token
   3.6 ICO Token

4. Conclusion
1. INTRODUCTION

1.1 How to Create Ethereum-based dApps

Ethereum dApps and Smart contracts are quickly becoming buzzwords in daily conversation, largely due to the rise of the blockchain technology and the promises it holds. As more and more companies and developers move to explore new business models out of blockchain, decentralized applications (dApps) and smart contracts are proving to be game changers and will surely disrupt enterprise operations very soon. With the introduction of a platform such as Ethereum new capabilities were added to enable smart contracts which could record and enforce agreements.

Smart contracts were first proposed in 1994 by Nick Szabo, an American computer scientist who invented a virtual currency called “Bit Gold” in 1998, full 10 years before the invention of Bitcoin. Szabo saw smart contracts as computerized transaction protocols that execute terms of a contract. He wanted to extend the functionality of electronic transaction methods, such as POS (Point of Sale), to the digital realm.
1.2 What is a Smart contract?

A **smart contract** refers to a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of computer code. The code and the agreements contained therein exist across a distributed, **decentralized blockchain network**. Smart contracts permit trusted transactions and agreements to be carried out among disparate, anonymous parties without the need for a central authority, legal system, or external enforcement mechanism. They render transactions traceable, transparent, and irreversible.

Smart contracts are shaping up to be the next big thing in the blockchain space with various start-ups moving in to become the best players when it comes to smart contract creation and adoption. In this publication, we would like to review how Ethereum-based **dApp Builder** seeks to bring smart contracts to the masses.
1.3 An Overview of dApp Builder and its Smart Contracts

Consider the case of an online store, where a business owner could easily create a decentralized application (dApp) on the dApp Builder platform without any coding skills.

Users can select Escrow smart contracts from the marketplace and use drag and drop functionality to quickly create Escrow dApps; and this will ensure that delivery confirmations and release of payments can be initiated and tracked over the blockchain.

This takes out the need for a third-party escrow service when facilitating peer-to-peer transactions.
2. Working with dApp Builder

2.1 Registration with dApp Builder

To begin using dApp Builder, please follow the steps shown:

1) Navigate to [https://dappbuilder.io/builder/](https://dappbuilder.io/builder/) and click on "Sign In" button on the top right corner. Fill in the form. We also use Google OAuth technology for registration on dApp Builder, so you can use your Google account to sign up. If you have already signed up with dApp Builder, please click on "Sign In" to return to the list of dApps you have already created.

2) After successful registration you will be automatically signed in and redirected to the Create New dApp page.
dApp Creation Form
2.2 Creating dApps with dApp Builder

On dApp creation form please indicate its name and choose a template (e.g. "Voting") of dApp from a dropdown list. Then you will have to fill out necessary fields dependent on the chosen type of your dApp.

Please note that to create a dApp you should install browser plugin MetaMask, create and unlock an account and choose Main Ethereum Network or Rinkeby Test Net as shown on the image below.

Our smart contracts are deployed in both this 2 networks. So you can firstly deploy your dApp in test network to try how it works before deploying it in main blockchain with real ETH.
Next, to complete the dApp creation transaction, you will have to pay for necessary Gas. In Rinkeby Test Net you can receive Ether required for such a transaction for free using Rinkeby Crypto Faucet.

After filling out the dApp creation form, press button "Create dApp" to publish your dApp to Ethereum blockchain and in MetaMask popup window confirm the gas payment transaction.

Then please wait till your transaction is mined and once mining is successful then soon you will be redirected to a page with a list of your dApps.

On that page you will be able to accomplish different changes of your decentralized applications.
2.3 Working with your dApps

You can see all of your created dApps on **My dApps** (for dApps deployed in Main Ethereum Blockchain) and **My Test dApps** (for dApps deployed in Rinkeby Test Network) pages.

**Created dApps**

© Copyright 2017 SIA "dApp Builder"
On this pages you can work with your dApp throw the web interface. For example, when it comes to a Voting dApp, you will be able to vote, see list of votes as well as finish the voting (provided that you have an active Ethereum account used during creation of the dApp).

Please note that in order to publish any smart contract changes in your dApps you will have to complete an Ethereum transaction and pay for the gas required by the transaction.
Also on this page you can get your **dApp web-link** or to share it in your socials, customize the look and feel of your dApps frontend.

For this please choose the dApp that you would like to customize the frontend of and press button **Customize the dApps Template**.
2.4 Working with your dApps on mobile, mobile dApp widget

To create a mobile application on Android on the basis of your dApp, please select a dApp and press the button Add selected dApp into Mobile App on My dApps page.

You will be redirected to a page with your mobile application where your dApp will be added as a widget. You can create several dApps and add all of them in a single mobile application.
For installing the mobile app on your Android device click on **Install Android App in 1 click!** button and get the installation link by **email, SMS** or **QR-code** scanning on your device.
3. Supported dApps

3.1 Voting

Voting dApp creation form
In dApp Builder the dApp creator customizes:

1. The list of options/candidates to vote for, giving them a name/description.
2. Whether or not the voting is "blind".

The dApp interface shows a list of configured candidates and click on a candidate name to record his/her vote. The vote of the user gets recorded in Ethereum blockchain through a WEBAPI call to a method of dApp Builder voting smart contract.

Also shown are:

1. If the voting is "blind" - how many votes are given for which candidate.
2. If the voting is not "blind" - how many votes are given for which candidate and by whom.

The dApp Creator can close the Voting when it should be finished.
3.2 Escrow

3.2.1 So how does the dApp Builder escrow dApp work?

Escrows on dApp Builder are executed using an open source Ethereum smart contract, typically an Escrow dApp creation process goes like this:

1) A buyer and seller confirm and agree on the terms of a sale of a seller's product. The buyer places Ethereum currency into the smart contract - this provides buyer's proof-of-funds to the seller.

2) The seller see that the buyer has enough funds and ships the product to the buyer producing an evidence of shipment to an independent third party called Escrow Agent.

3) Escrow Agent sees the evidence of shipment and initiates the transfer of the funds stored in the escrow to the seller. The transaction is now complete.

4) In case if the seller does not produce an evidence of shipment to the Escrow Agent then Escrow Agent returns the funds from escrow to the buyer. Buyer receives her money.

5) Escrow Agent can get a small percentage of the transaction amount for her impartial arbitration.

Smart contracts will enable users to safely exchange ether/bitcoin with one another and in case of a dispute they will be able to name a trusted third-party to mediate a trade. dApp Builder will currently act as the mediator in case of disputes.
3.2.2 Creating and Funding an Escrow

When a user (say a buyer) wants to trade with somebody else on the dApp platform, a buyer will have to initiate the trade and fund it in a single transaction. Thereafter an escrow will be created on the blockchain. Every escrow created will first require a signed invitation from dApp Builder; this is just a way to keep the contract clean. A seller can request one of these signatures from dApp Builder’s API whenever they are ready to place their ether in escrow.

The temporary invitation contains a signature of the trade’s properties, including:

1) The seller’s address used to interact with the escrow and receive funds.
2) The buyer’s address used to interact with the escrow, and receive the returned ether in case of cancellation.
3) The size of the trade in ether.
4) dApp Builder’s commission.
5) The payment window in seconds (except for cash trades).

Funding an escrow is easy as sellers can either choose their encrypted in-browser wallet with one-click or use the other option of simply copying the necessary data value to initiate the escrow from an external wallet.
The gas cost for creating an escrow is paid by the seller and he/she is free to choose a gas price that they are comfortable with. The createEscrow function uses approximately 69,000 gas – which costs less than US10 cents.

Before you fund an escrow, it is imperative to ensure that both you as the seller and the buyer are in total agreement in regards to the terms, as it costs a few cents to fund an escrow even if you end up cancelling it. And to avoid needlessly locking up your ether, sellers should wait until the buyer responds with the agreement of the terms.
Once the initial **escrow creation** transaction has been confirmed by the network, the escrow will now exist on the blockchain and can easily be verified by anyone.

And once **dApp Builder** confirms that the escrow has been created and funded, which normally takes a few seconds, the trade will appear like this:
3.2.3 How about making changes to an escrow?

Once the escrow has entered this stage where the buyer has essentially locked the ether in escrow, there are three ways the escrow can end:

1) The buyer can release the funds to the seller.
2) The buyer can cancel the trade.
3) Either party can call in the arbitrator to resolve the escrow.

dApp Builder uses smart contracts to bring blockchain technology to the real business world so as to cover several use cases such as payments, insurance, legal, etc. The bottom line is that bringing dApps to the masses will streamline transactions and as a result minimize friction and improve customer experiences both for B2C and B2B companies.
3.3 Multisignature Wallet

In dApp Builder the dApp creator customizes:

1) initial balance of the wallet;
2) the number of needed approvals for sending transactions;
3) the wallet’s owners (a list of ETH accounts which can approve transactions).

Multisignature Wallet dApp creation form
The dApp interface shows the wallet information (dApp name, owners, needed approvals, balance), a list of sent and unsent transactions. Each dApp user can send ether to the wallet through the special form.

The wallet’s owner can initialize a new transaction, approve an existent transaction or send an already approved transaction.

---

**Multisignature Wallet interface**

**Sending ETH to wallet**

**Creating a new transaction**

**Unconfirmed transaction**
3.4 Betting

The application creator specifies the name of the event, fills the list of contenders, appoints an arbitrator and indicates the percentage of the arbitrator's share from the winnings.
All participants can bet any amount in ETH on any candidate. When the event occurred, the arbitrator stops taking bets and announces a winner when he is defined. The funds bet on the losers (except the percentage of the arbitrator's share) form a prize fund.
Then, those who bet on the winner may pick up their winnings. It consists of their bets and their share in the prize fund, proportional to the amount of funds bet.

For example, if two people bet on the winner: the first one had put 1 ETH, and the another one - 2 ETH, the second player will receive the winnings twice more than the first one's.

To collect the your winnings, you must click the Withdraw button at the dApp interface. The arbitrator gets his share the same way.

If no one has bet on the winner, all players can reclaim their bet (minus those funds that went to pay for the arbitrator's services).

In addition to bidding, the user can view his bets, as well as his balance, which he can withdraw from the smart contract, if the winner has been already defined.
3.5 Custom token

You can create your own **ERC20** token by specifying the standard parameters, such as **token name**, **token symbol**, **decimals number**, **total supply** and the **address** of the initial holder of all the tokens.
dApp Builder creates your own smart contract (you can see the Solidity code here) for your token and an interface for all the **token operations**:

1) **Transfering**: each token holder can transfer a specified number of his/her tokens to another Ethereum address.

2) **Checking the token balance** of any Ethereum address.
3) **Approving:** each token holder can allow another person or smart contract to use a specified value of his/her tokens. Also token holder can check and change his/her current token allowance.

4) **Transferring from another address:** if someone allowed you to transfer his/her tokens you can also do it by using dApp interface.
You can customize the **colourscheme** of your token interface. And use it both on your desktop with **MetaMask** and on your **Android** mobile device with dApp Widget.
3.6 ICO Token

You can create your own ICO token based on ERC20 standard. It’s an extended version of Custom Token dapp that we have presented earlier. You can read the solidity code here.

A dApp creator specifies the standard token parameters (such as token name, token symbol and decimals number) and the special parameters for token sale:

1. **Initial supply** — number of tokens that will be issued immediately, before the token sale starts.
2. **Token fund deposit** — an Ethereum address that get initial token supply.
3. **ETH deposit** — an Ethereum address that will get Ether after ICO successfully finish.
4. **Token exchange rate** — how many tokens a customer can buy for 1 ETH.
5. **Token creation cap** — the maximum number of tokens for issue.
6. **Token creation min** — the number of tokens that must be issued before the ICO ends.
7. **Start funding block** — the Ethereum block number after which ICO starts.
8. **End funding block** — the Ethereum block number after which ICO stops.
Then the creation form is submitted and smart contract creating transaction is successfully mined the dApp creator gets his/her own smart contract for ICO and token fund deposit gets the initial token supply.

The token sale starts after the start funding block and goes on before the end funding block or before the token creation cap is reached.
The main dApp screen is the ICO screen:
Here customers can buy tokens and watch the token sale progress. A **customer** can enter the number of tokens he/she wants to buy or the value of Ether he/she wants to spend (the number of tokens will be calculated automatically).
In the **sidebar menu** user can switch into token screen. This is the screen for standard token operations like the interface for **Custom Token dApp**, that we have made earlier.

**ICO Token dApp:**

*Token interface*
After the **successfully finished ICO** the organizer (ETH deposit) can withdraw all ETH from the contract.
If ICO ends before the reaching of token creation min, the ICO organizer can not withdraw Ether from contract and all the customers (except for holder of initial supply) can do refund.

ICO Token dApp: Refunding
4. CONCLUSION

The dApps Builder Voting, Escrow and MultiSig Wallet smart contracts described above is just the start of the long list of contracts dApps Builder is planning to develop.

In addition to expanding the number of smart contracts for different business use cases, we are planning to:

1) add more options and configurability to smart contracts that already exists
2) provide an independent security audit of the contracts code
3) improve user experience and make creating new smart contracts easier

For more information, please follow the