PeepEth iOS Client

Authors

Georgy Fesenko

fesenko.g@gmail.com

Anton Grigoriev

antongrigorjev2010@gmail.com

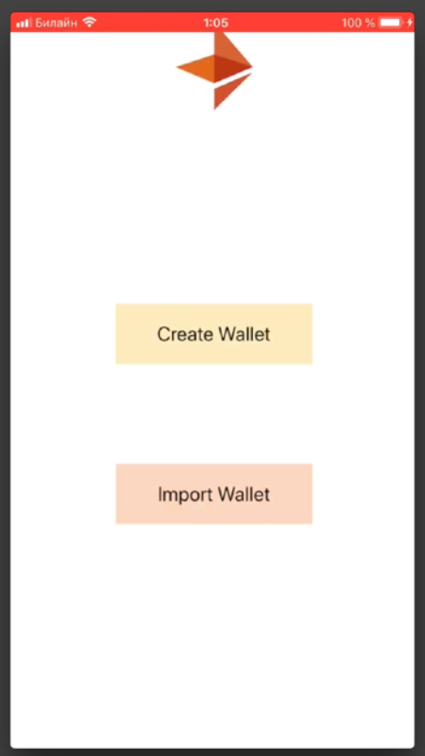
PeepEth is a decentralized alternative to Twitter. Each peace of data is stored in IPFS and the Ethereum blockchain.

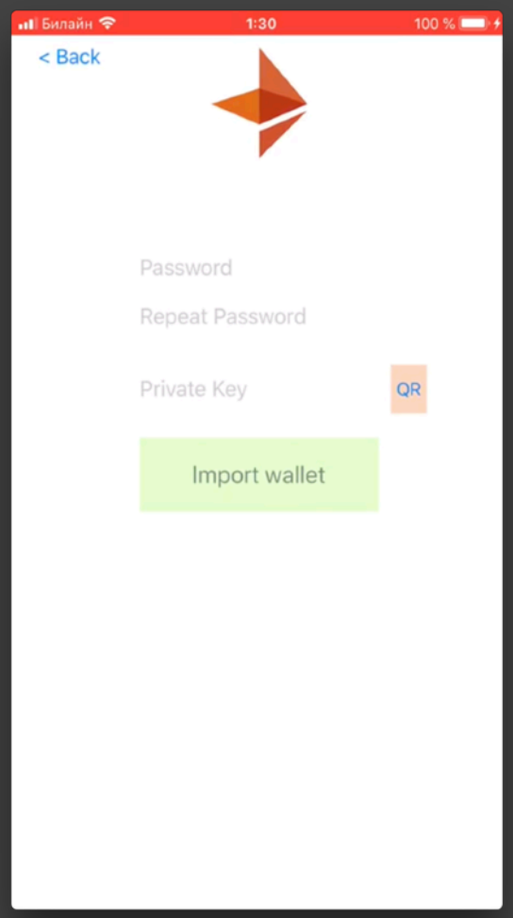
The purpose of that app is to show how easy and fast it is(three days of hackathon) to implement blockchain-based DApp with contract-interaction functionality on iOS platform

Our app provides the following functionality

Import or create a wallet

Which will be saved securely in the local storage of your device

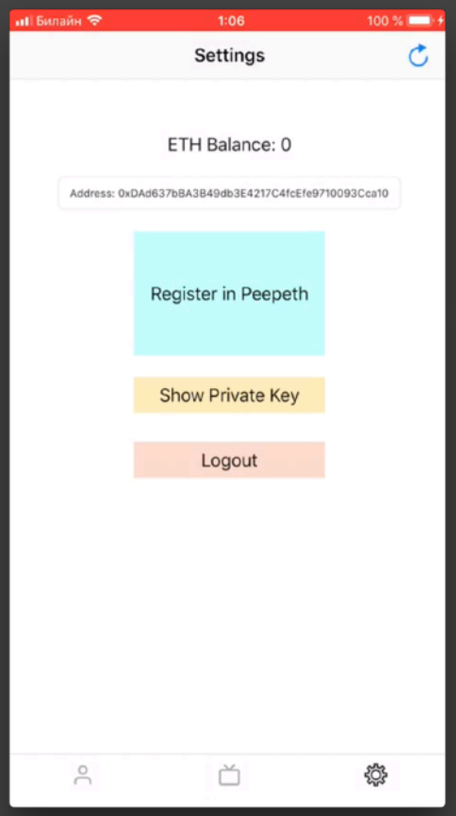


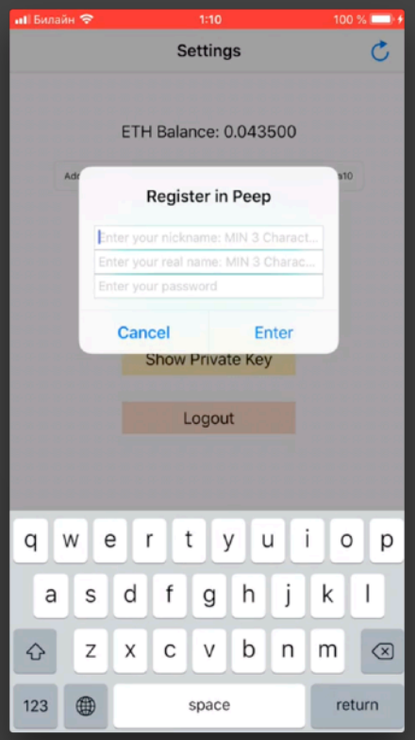


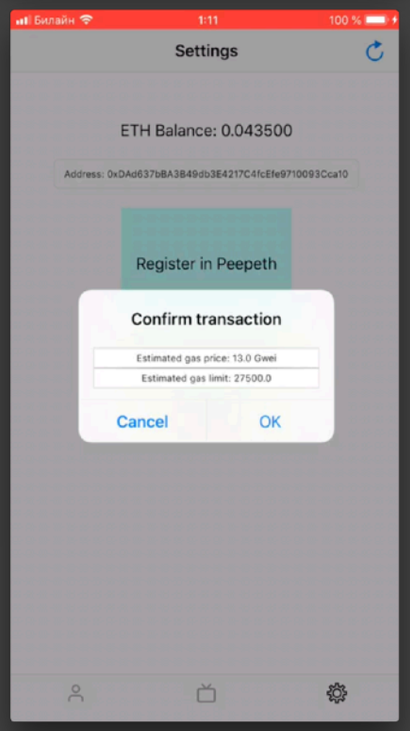

```
func addNewWalletWithPrivateKey(key: String, password: String, completion: @escaping (KeyWalletModel?, Error?)
    -> Void) {
    let text = key.trimmingCharacters(in: .whitespacesAndNewlines)
    guard let data = Data.fromHex(text) else {
        completion(nil, WalletSavingError.couldNotSaveTheWallet)
        return
    }

    guard let newWallet = try? EthereumKeystoreV3(privateKey: data, password: password) else {
        completion(nil, WalletSavingError.couldNotSaveTheWallet)
        return
    }

    guard let wallet = newWallet, wallet.addresses?.count == 1 else {
        completion(nil, WalletSavingError.couldNotSaveTheWallet)
        return
    }
    guard let keyData = try? JSONEncoder().encode(wallet.keystoreParams) else {
        completion(nil, WalletSavingError.couldNotSaveTheWallet)
        return
    }
    guard let address = newWallet?.addresses?.first?.address else {
        completion(nil, WalletSavingError.couldNotSaveTheWallet)
        return
    }
    let walletModel = KeyWalletModel(address: address, data: keyData)
    completion(walletModel, nil)
}
```

Register an account

Sending a transaction using web3swift library



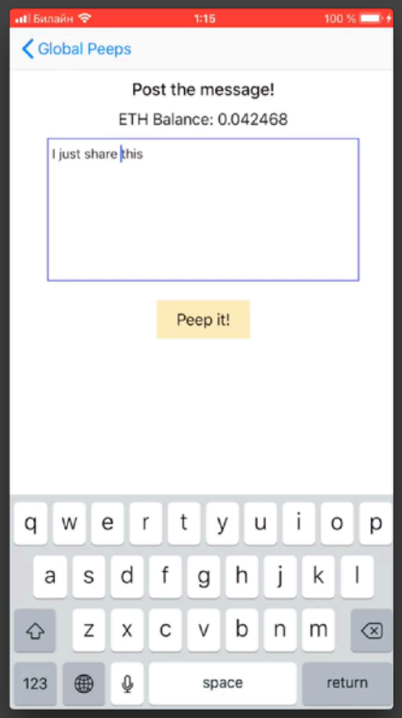


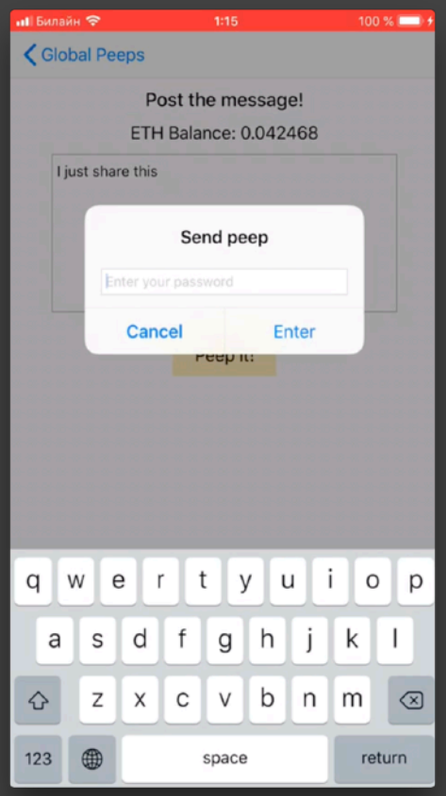


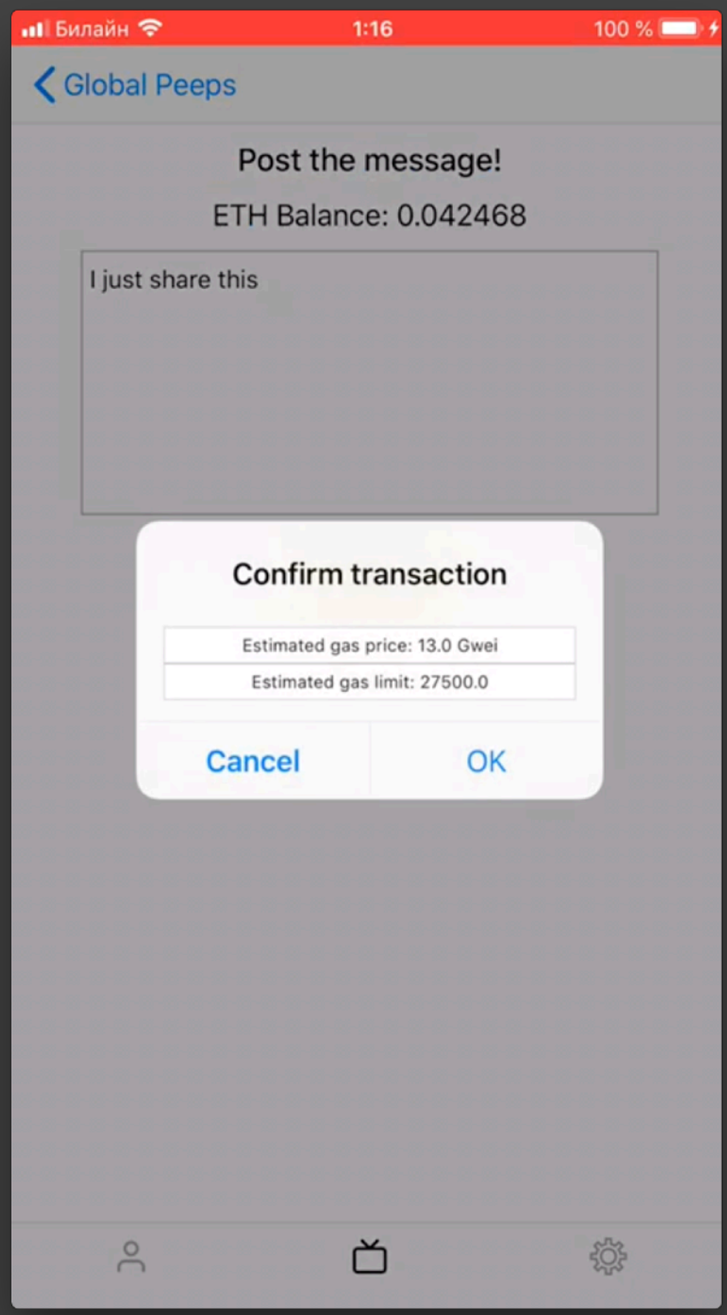

```swift
//MARK: - Account creation
func prepareCreateAccountTransaction(username: String, userDataHash: String, gasLimit: BigUInt = 27500,
    completion: @escaping (Result<TransactionIntermediate>) -> Void) {
    //preparing
    DispatchQueue.global().async {
        let wallet = self.keyservice.selectedWallet()
        guard let address = wallet?.address else { return }
        let ethAddressFrom = EthereumAddress(address)

        let web3 = Web3.InfuraMainnetWeb3()
        web3.addKeystoreManager(self.keyservice.keystoreManager())
        guard let contract = web3.contract(peepEthABI, at: ethContractAddress, abiVersion: 2) else { return }

        var options = Web3Options.defaultOptions()
        options.from = ethAddressFrom
        options.value = 0
        options.gasLimit = gasLimit

        guard let gasPrice = web3.eth.getGasPrice().value else { return }
        options.gasPrice = gasPrice
        print(gasPrice)

        guard let transaction = contract.method("createAccount", parameters: [username, userDataHash] as
            [AnyObject], options: options) else { return }
        print(transaction.transaction.data)
        let con = ContractV2(peepEthABI)
        print(transaction.transaction.data.toHexString())
        if let input = con?.decodeInputData(transaction.transaction.data) {
            print(input)
        }
        guard case .success(let estimate) = transaction.estimateGas(options: options) else {return}
        print(estimate)
        DispatchQueue.main.async {
            completion(Result.Success(transaction))
        }
    }
}
```

Sending a peep

Sending a transaction using web3swift library



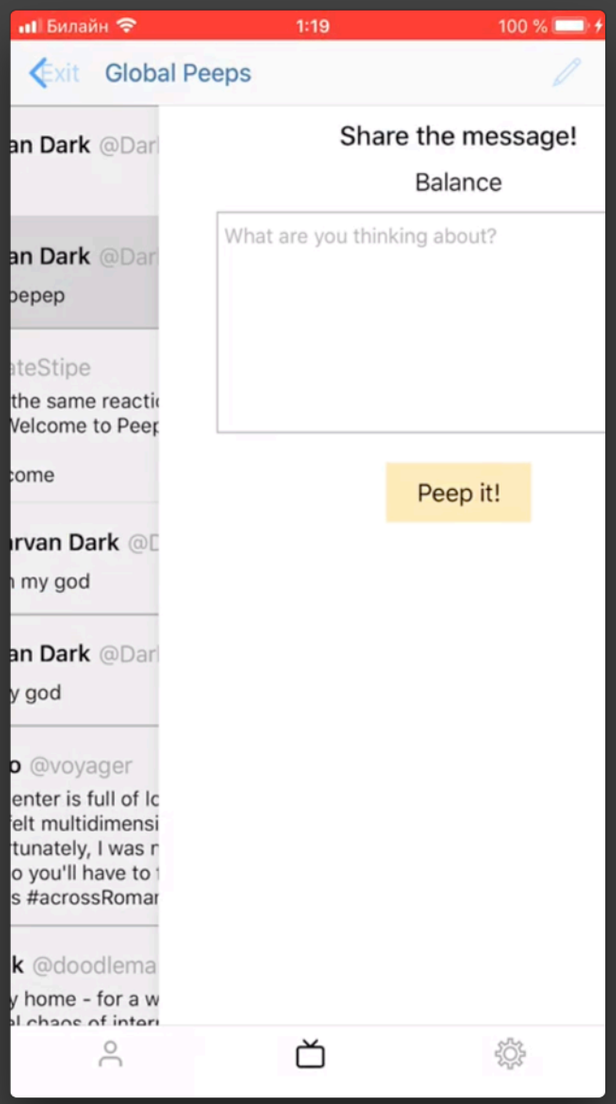





```
//MARK: - Posting a peep
func preparePostPeepTransaction(peepDataHash: String, gasLimit: BigUInt = 27500 ,completion: @escaping
    (Result<TransactionIntermediate>) -> Void) {
    DispatchQueue.global().async {
        let wallet = self.keyservice.selectedWallet()
        guard let address = wallet?.address else { return }
        let ethAddressFrom = EthereumAddress(address)

        let web3 = Web3.InfuraMainnetWeb3()
        web3.addKeystoreManager(self.keyservice.keystoreManager())

        var options = Web3Options.defaultOptions()
        options.from = ethAddressFrom
        options.value = 0
        guard let contract = web3.contract(peepEthABI, at: ethContractAddress, abiVersion: 2) else { return }
        guard let gasPrice = web3.eth.getGasPrice().value else { return }
        options.gasPrice = gasPrice
        options.gasLimit = gasLimit
        guard let transaction = contract.method("post", parameters: [peepDataHash] as [AnyObject], options:
            options) else { return }
        guard case .success(let estimate) = transaction.estimateGas(options: options) else {return}
        print("estimated cost: \(estimate)")
        DispatchQueue.main.async {
            completion(Result.Success(transaction))
        }
    }
}
```

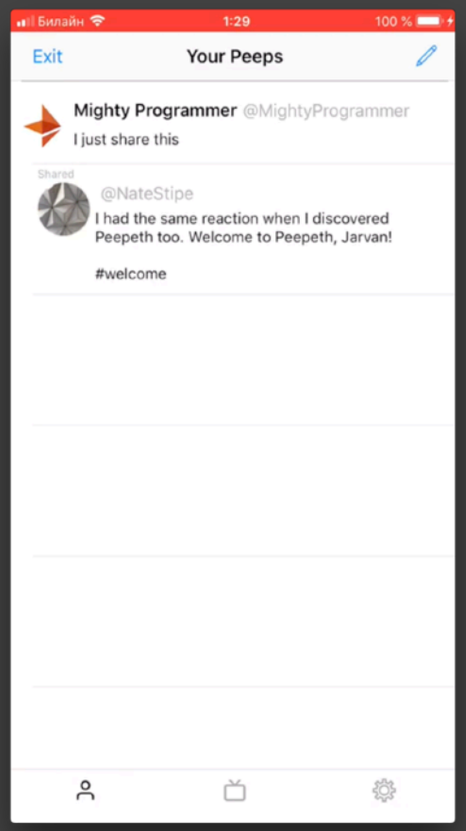
Share peep of other user

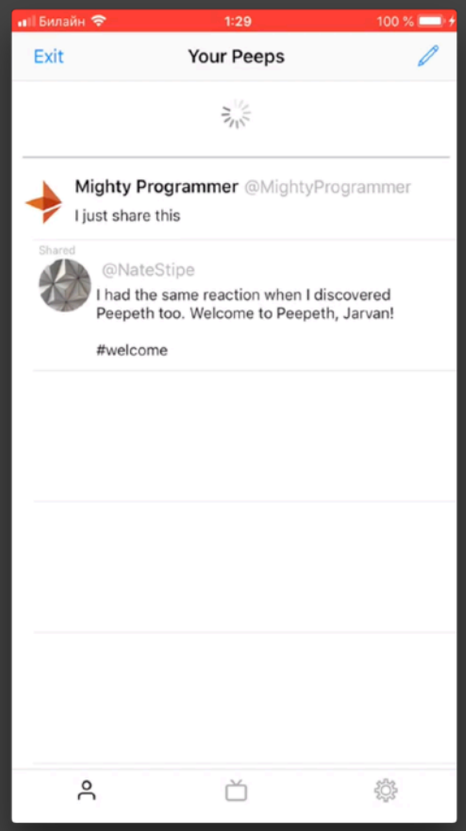
Sending a transaction using web3swift library

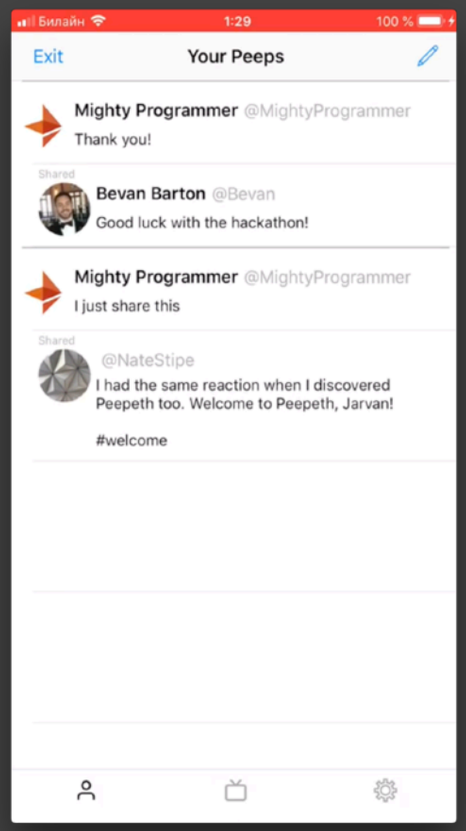


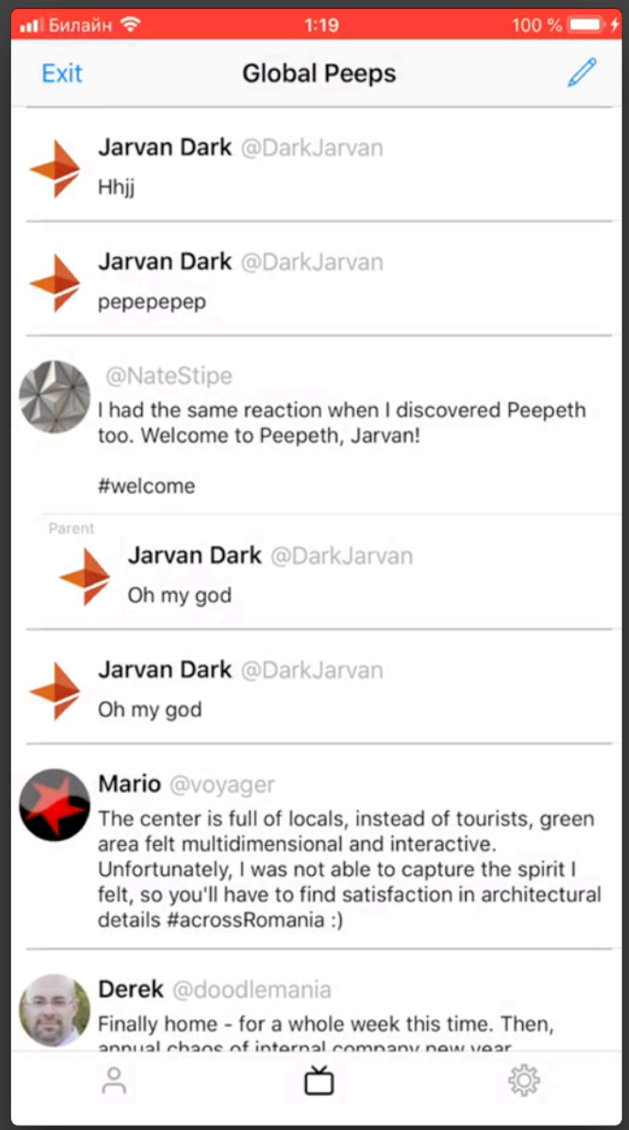

```
func prepareSharePeepTransaction(peepDataHash: String, gasLimit: BigUInt = 27500, completion: @escaping
    (Result<TransactionIntermediate>) -> Void) {
    DispatchQueue.global().async {
        let wallet = self.keyservice.selectedWallet()
        guard let address = wallet?.address else { return }
        let ethAddressFrom = EthereumAddress(address)

        let web3 = Web3.InfuraMainnetWeb3()
        web3.addKeystoreManager(self.keyservice.keystoreManager())

        var options = Web3Options.defaultOptions()
        options.from = ethAddressFrom
        options.value = 0
        guard let contract = web3.contract(peepEthABI, at: ethContractAddress, abiVersion: 2) else { return }
        guard let gasPrice = web3.eth.getGasPrice().value else { return }
        options.gasPrice = gasPrice
        options.gasLimit = gasLimit
        guard let transaction = contract.method("share", parameters: [peepDataHash] as [AnyObject], options:
            options) else { return }
        guard case .success(let estimate) = transaction.estimateGas(options: options) else {return}
        print("estimated cost: \(estimate)")
        DispatchQueue.main.async {
            completion(Result.Success(transaction))
        }
    }
}
```

Upload either global or personal peeps history









We hope that we will inspire somebody on that planet to try their best in blockchain!

We want to thank Bevan Barton for help with PeepEth and, of course, BANKEX Foundation team for their open-source library!

Best wishes and thanks for such an opportunity :)