PROTOFIRE

Developer experience, from Ethereum to Tezos through deFi

Lend & Borrow

April, 23rd, 2020

Mariano Agüero
Sr. blockchain Engineer

Nicolás Dominguez
Sr. blockchain Engineer

Cristian Malfesi
Project Manager

Milestones

#2

Interaction between smart-contracts
FA1.2 Pool Implementation
Security Mechanisms

#3

Borrow & Repay

Integration between the Pool contract and the FA1.2 contract

PROTOFIRE

Some Defi concepts used in the contracts

The Collateral Factor

The maximum amount users can borrow is limited by the collateral factor. Right now is 80%.

Account Liquidity

The pool contract provides an easy to use function that calculates your account's liquidity,

Borrow Balance

This is the sum of a user's current borrowed amount plus the interest that needs to be repaid

Interest Rate Model

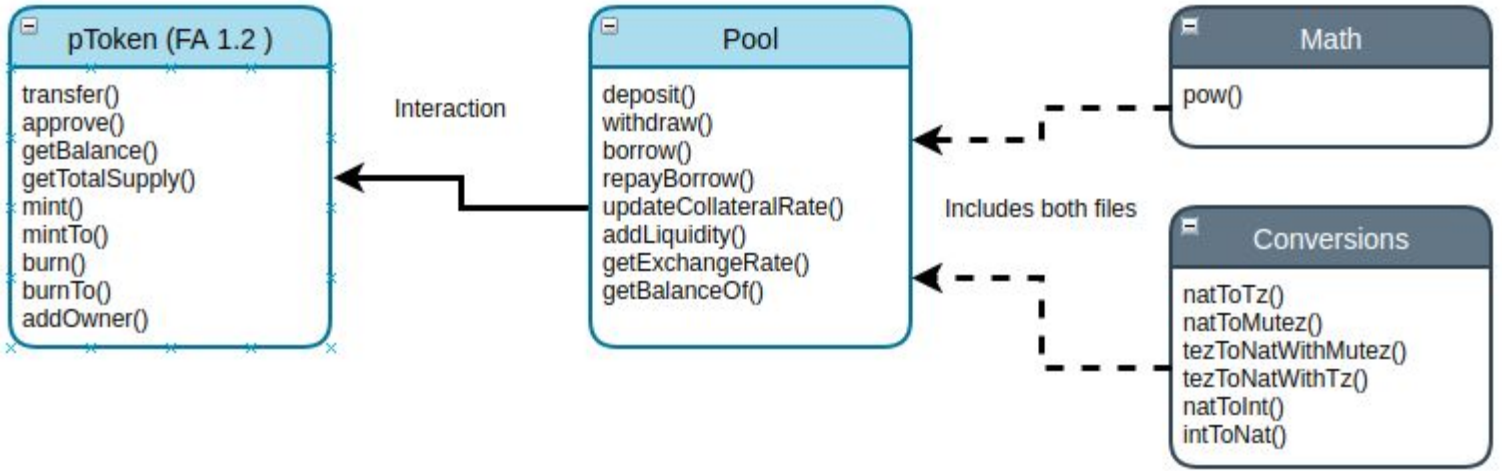
Following the "Price Theory", interest rate will act as a function of demand and supply, resulting in a decrease in interest rate when the demand is low and vice versa when the demand is high.

Based on:

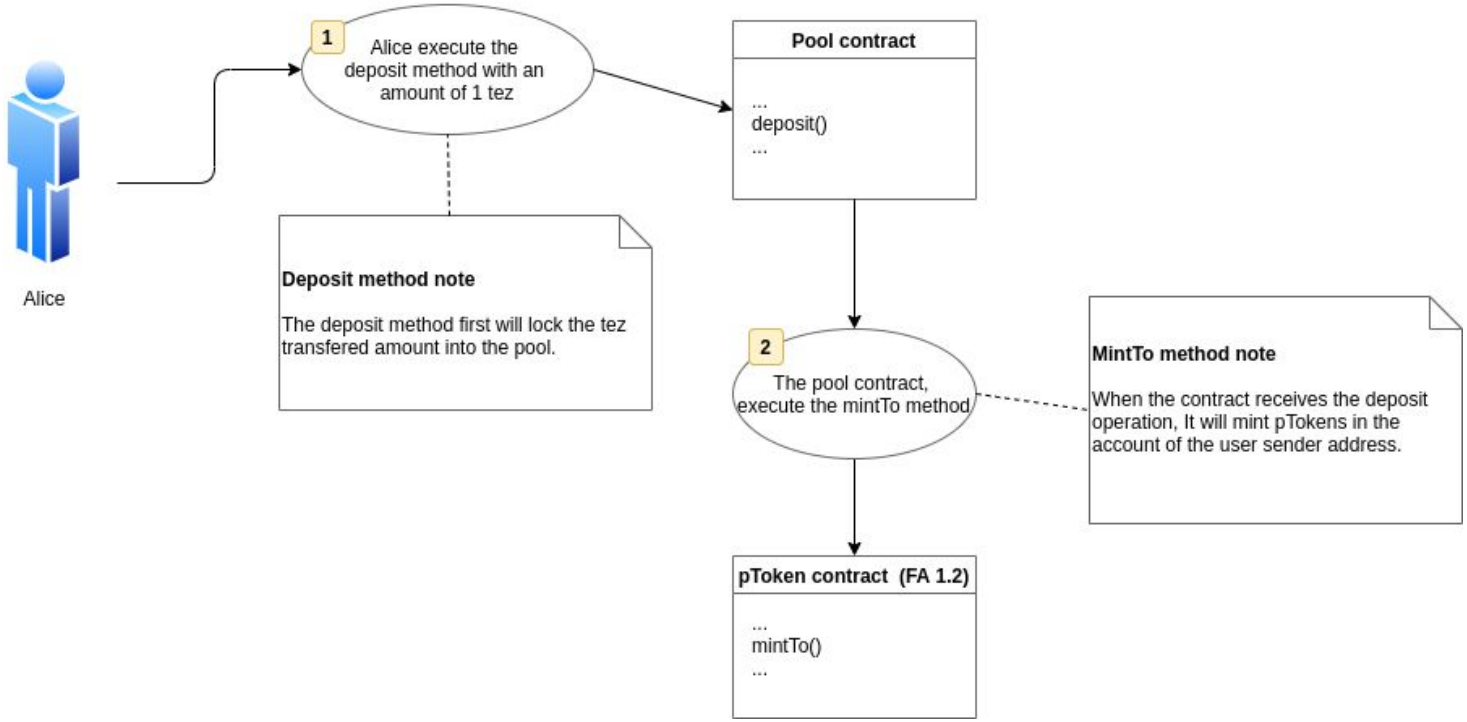
- Compound whitepaper
- Compound documentation
- Compound protocol
- Compound blogs, supply assets and borrow assets

PROTOFIRE

Contracts architecture

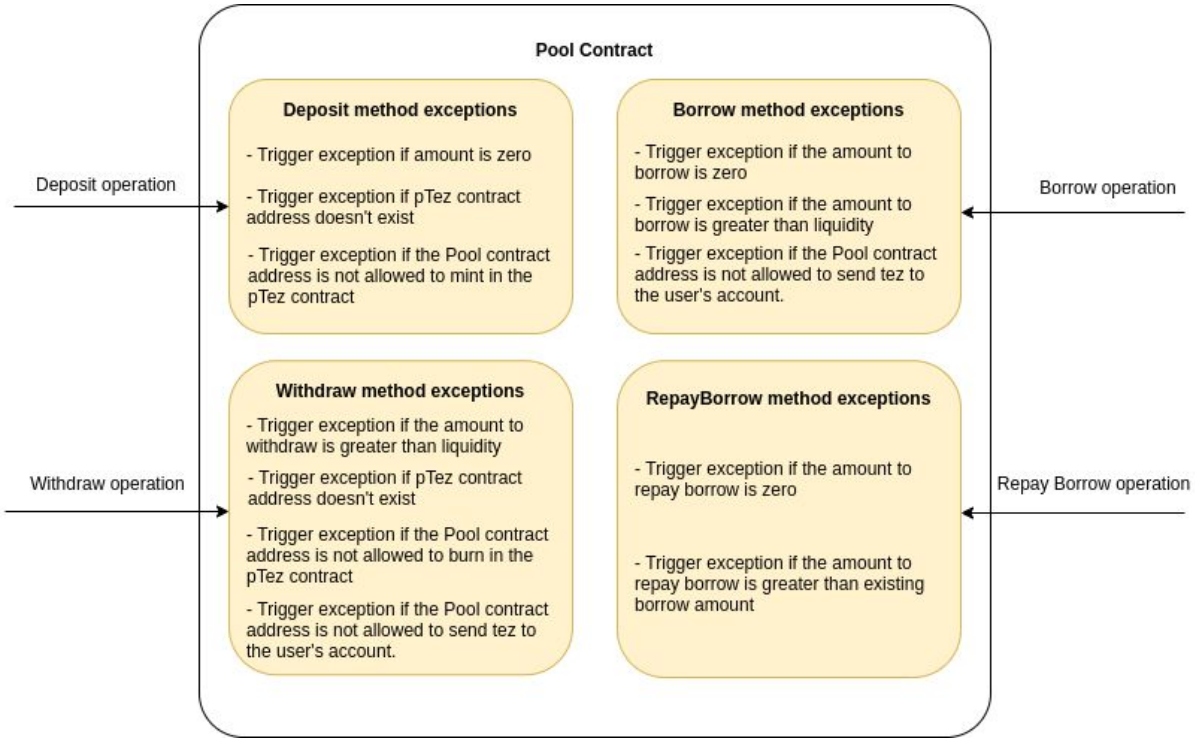


Deposit method & interactions contracts



PROTOFIRE

Pool contract security



Number of exceptions

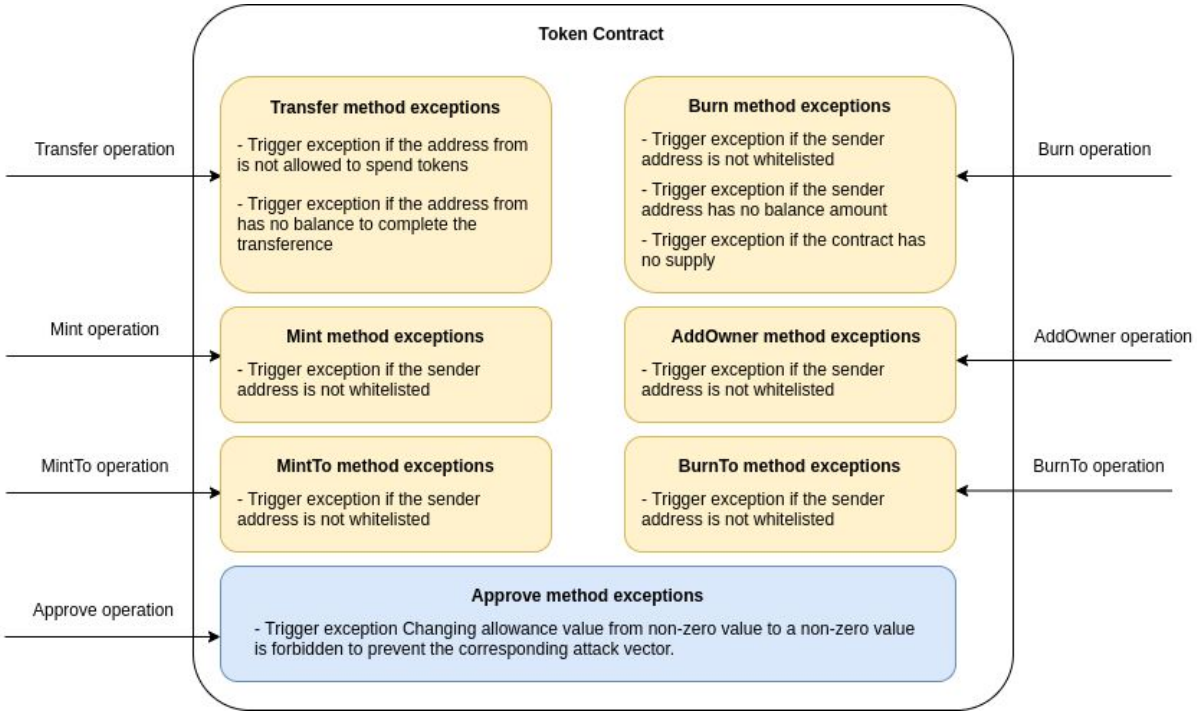
Business rules: 3

Inter-contract invocation: 2

Basic contract validation: 7

PROTOFIRE

Token contract security



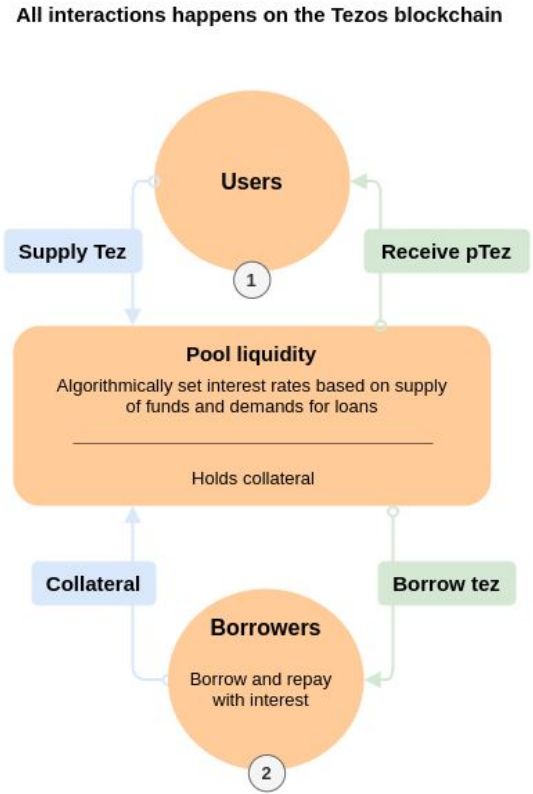
Number of exceptions

Business rules: 5

Basic contract validation: 5

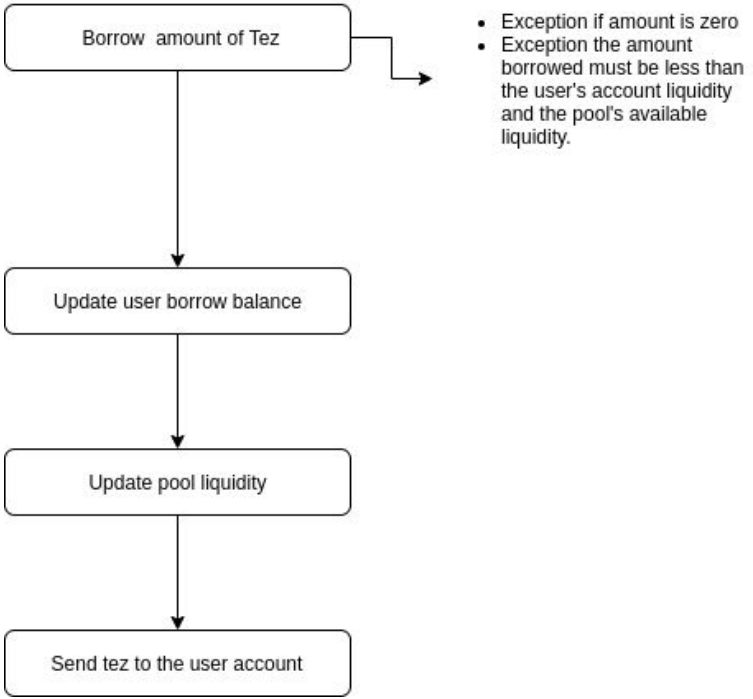
PROTOFIRE

Interactions model



Borrow

Borrow method - Step by Step



- Exception if amount is zero
- Exception the amount borrowed must be less than the user's account liquidity and the pool's available liquidity.

Pool contract

borrows : []

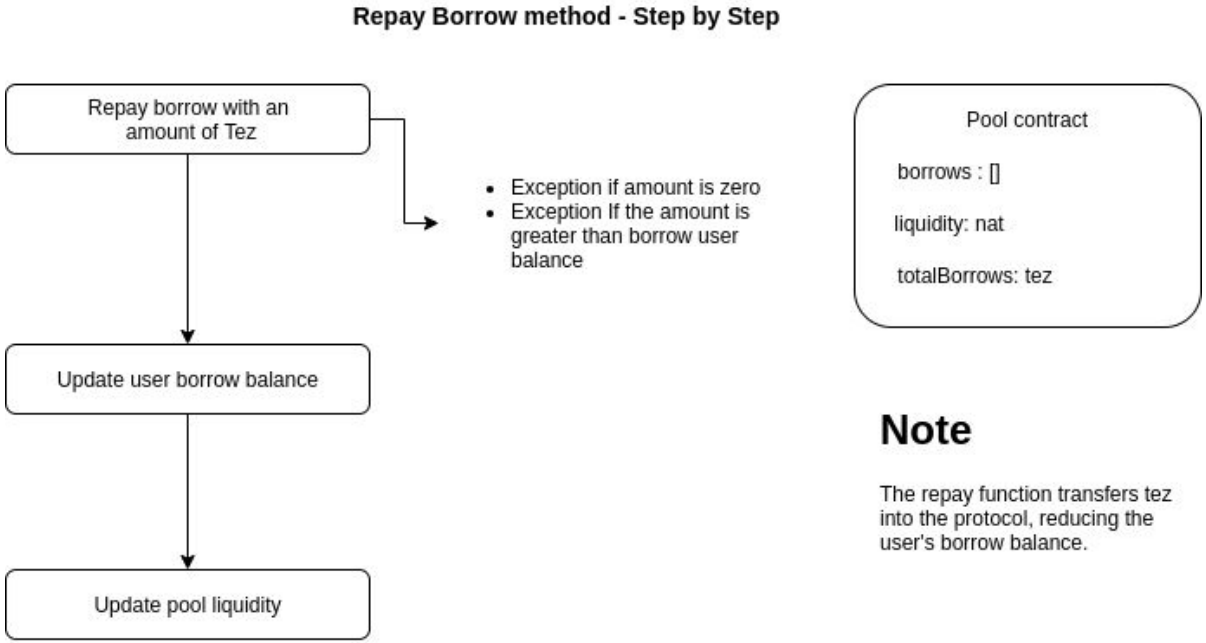
liquidity: nat

totalBorrows: tez

Note

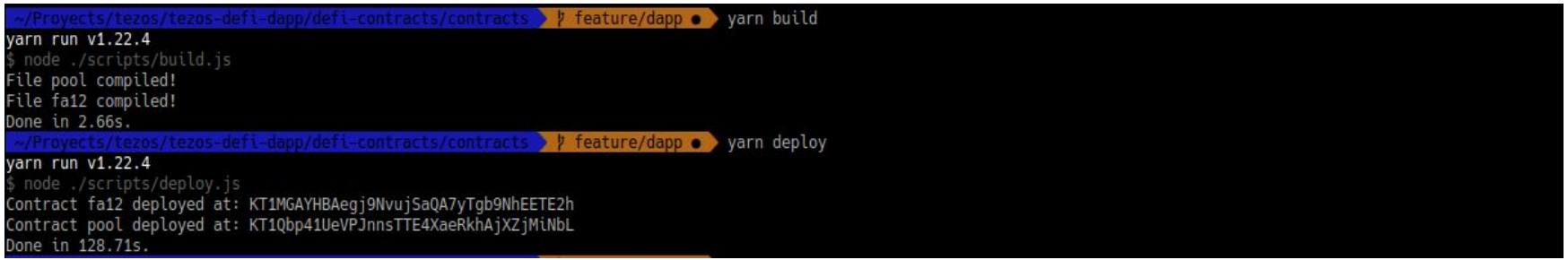
The borrow function transfers tez from the protocol to the user, and creates a borrow balance which begins accumulating interest. The amount borrowed must be less than the user's account liquidity and the pool's available liquidity.

Repay borrow



PROTOFIRE

Demo



Repository: https://github.com/protofire/tezos-defi-dapp

PROTOFIRE

Demo

```
~/Proyects/tezos/tezos-defi-dapp/defi-contracts  feature/dapp ●  yarn example:deposit
yarn run v1.22.4
$ node ./examples/deposit.js
```

Action	Account address	Account balance	Fee	Pool: deposit account balan…	Pool: total depos…	Token: account ba…
Before deposit 40…	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19106.775466 tz	0 tz	0 tz	0 tz	0.00 ptez
After deposit 40 tz	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19066.448801 tz	0.089957 tz	40 tz	40 tz	40.00 ptez

```
Check pool contract transactions 'https://better-call.dev/carthage/KT1PVuhe9QaXuCacKnQUH2BweQ9kLHBnwKc5/operations'
Check token contract transactions 'https://better-call.dev/carthage/KT1CJQWBHP8HTF4rddcQek9uMMfNGeeCmRuM/operations'
Done in 115.29s.
```

```
~/Proyects/tezos/tezos-defi-dapp/defi-contracts  feature/dapp ●  yarn example:withdraw
yarn run v1.22.4
$ node ./examples/withdraw.js
```

Action	Account address	Account balance	Fee	Pool: deposit account balan…	Pool: total depos…	Token: account ba…
Before withdraw 1…	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19066.448801 tz	0 tz	40 tz	40 tz	40.00 ptez
After withdraw 1 tz	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19067.253647 tz	0.092384 tz	39 tz	39 tz	39.00 ptez

```
Check pool contract transactions 'https://better-call.dev/carthage/KT1PVuhe9QaXuCacKnQUH2BweQ9kLHBnwKc5/operations'
Check token contract transactions 'https://better-call.dev/carthage/KT1CJQWBHP8HTF4rddcQek9uMMfNGeeCmRuM/operations'
Done in 74.61s.
```

Repository: https://github.com/protofire/tezos-defi-dapp

PROTOFIRE

Demo

```
~/Proyects/tezos/tezos-defi-dapp/defi-contracts  feature/dapp ●  yarn example:borrow
yarn run v1.22.4
$ node ./examples/borrow.js
```

Action	Account address	Account balance	Fee	Pool: deposit bal…	Pool: borrow bala…	Pool: total depos…	Pool: total borro…
Before borrow 1 tz	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19067.253647 tz	0 tz	39 tz	0 tz	39 tz	0 tz
After borrow 1 tz	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19068.199723 tz	0.049924 tz	39 tz	1 tz	39 tz	1 tz

```
Check pool contract transactions 'https://better-call.dev/carthage/KT1PVuhe9QaXuCacKnQUH2BweQ9kLHBnwKc5/operations'
Done in 38.57s.
```

```
~/Proyects/tezos/tezos-defi-dapp/defi-contracts  feature/dapp ●  yarn example:repayborrow
yarn run v1.22.4
$ node ./examples/repayBorrow.js
```

Action	Account address	Account balance	Fee	Pool: deposit bal…	Pool: borrow bala…	Pool: total depos…	Pool: total borro…
Before repay borr…	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19068.199723 tz	0 tz	39 tz	1 tz	39 tz	1 tz
After repay borro…	tz1V3KPWxv4gqFrVr1wuoW8kD5JSFcRsWapb	19067.151852 tz	0.047871 tz	39 tz	0 tz	39 tz	0 tz

```
Check pool contract transactions 'https://better-call.dev/carthage/KT1PVuhe9QaXuCacKnQUH2BweQ9kLHBnwKc5/operations'
Done in 41.10s.
```

Repository: https://github.com/protofire/tezos-defi-dapp

Blockers, Issues

Size of the pool contract

The allowed size is 16kb , in ethereum is 24kb. There is a very limited documentation and examples for inline attributes.

Math operations

Use of conversions functions, increase the contract size. Probably it is a good option that ligo provides some math functions as a external library or maybe as a part of the language.

Ligo error messages

They are not very friendly. Sometimes error line is not mentioned, or in which function the problem is located.

Exceptions

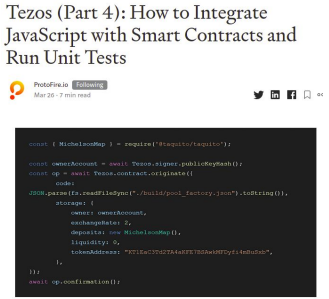
It would be nice to pass some parameters to the 'failwith' method. Will improve the dev and user experience. Maybe the failwith function could be improved in the ligo language, and allow us to interpolate error messages.

View methods

It would be nice to have a 'view' method in the contracts. From a contract, it is not possible to have an entrypoint which returns a data (after a computation or from the storage) outside the contract. Exist this TZIP.

PROTOFIRE

Publications

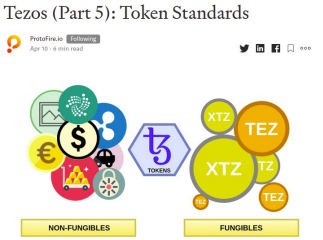


Tezos (Part 4): How to Integrate JavaScript with Smart Contracts and Run Unit Tests

https://medium.com/protofire-blog/tezos-part-4-how-to-integrate-javascript-with-smart-contracts-and-run-unit-tests-c36756149e9d

365 claps

34 Retweets **58** Likes



Tezos (Part 5): Token Standards

https://medium.com/protofire-blog/tezos-part-5-token-standards-28b8733a3ce5

122 claps

47 Retweets **65** Me gusta

Next blog posts

- Tezos (Part 6): FA 1.2 specification and interactions between contracts
- Tezos (Part 7): Borrow & Repay - Tips to improve the experience
- Tezos (Part 8): Integrate a user interface with Tezos smart contracts

PROTOFIRE

Next steps

#1 **User Interface implementation with Taquito**

#2 **Lend and Borrow integration as backend**

#3 **Blog post: “How to build a UI and interact with Tezos smart-contracts”**

PROTOFIRE

Questions?...

https://protofire.io

@ProtoFire_io @protofire_io Protofire-io https://github.com/protofire