
Petyl Documentation

Release 0.0.7

Adrian Guerrero

Jul 24, 2020

Contents

1	Petyl	1
1.1	Features	1
2	Petyl Protocol	3
2.1	Petyl Framework	3
3	Tokens	5
3.1	Creating a New Token	5
4	Partitions	7
4.1	Hybrid tokens	7
4.2	Token types	7
4.3	Partitions ID	7
5	Converters	9
5.1	Use Cases	9
5.2	Adding Converters	9
5.3	Converting Tokens	9
6	Token Rules	11
6.1	Examples	11
7	Whitelists	13
8	Documents	15
9	Token Types	17
10	Base Tokens	19
10.1	Functions	19
10.2	Send vs Transfer	19
10.3	Transfer with data	19
10.4	Operators	19
10.5	ERC1820 Registry	20
11	Distribution Tokens	21
11.1	Functions	21
11.2	Deposit Dividends	21

11.3	Withdraw Funds	21
12	Ventures	23
12.1	Components	23
12.2	Partitions	23
12.3	Token Converters	23
12.4	Documents	23
13	Options	25
14	Dutch Auctions	27
14.1	How a Dutch Auction Works	27
14.2	Set Up	27
14.3	More information	27
15	Links	29

Petyl is a token framework for revenue generating ventures

!! Documentation is currently under construction !!

Note: This project relies upon some knowledge about `Solidity` and the documentation assumes a basic familiarity with it. You may wish to view the [Solidity docs](#) if you have not used it previously.

1.1 Features

- Hybrid token structure - distinct asset classes combined with fungible tokens
- Token converters for converting between different classes
- Customisable rule engine for token transfers
- Different token types - Dividend tokens, vested tokens, options etc
- Contract testing via [pytest](#), including trace-based coverage evaluation
- Property-based and stateful testing via [hypothesis](#)

CHAPTER 2

Petyl Protocol

Petyl is designed to help value generating ventures grow.

It came from designing a collection of tokens for different purposes - yet all related to the same project. As the project grew, there were different types of incentives that were proposed to help us grow. Ultimately, we tokenised each of the incentives, and wrote some smart contracts with rules for redeeming each of the incentives. It has since been generalised to support a number of token types and use cases.

2.1 Petyl Framework

Petyl can best be described as a token framework for managing a collection of tokens relating to a value generating entity. We'll call that entity a Tree

This page provides a quick overview of Petyl. It relies mostly on examples and assumes a level of familiarity with smart contract development. For more in-depth content, you should read the documentation sections under “Getting Started” in the table of contents.

If you have any questions about the Petyl protocol, feel free to ask on [Ethereum StackExchange](#) or join us on [Gitter](#).

3.1 Creating a New Token

You can visit the “[Petyl Venture Studio](#)”,

The Petyl protocol is a framework for building hybrid tokens. One such token type is the Hybrid token.

4.1 Hybrid tokens

A hybrid token is a combination of distinct assets, like an ERC721, and fungible assets such as an ERC20. One common example is for company shares, where they all relate to ownership of the company, but different classes of shares, say founder shares have different conditions than regular shares.

We can build hybrid tokens with a main contract, the Petyl contract, with one or many partitions. Partitions are separate tokens which relate to the same underlying asset but distinct from the different partitions.

4.2 Token types

Both the base token type and the dividend tokens can be added as partitions within the main Petyl contract.

4.3 Partitions ID

Each base token within the petyl protocol has a Partition ID when created. This is based on a hash and unique to each token deployed.

Converters can be connected to the main petyl contract. Depending on the tokens, the converter is able to mint and burn from one or more token contracts. This allows for both the creation and conversion of tokens.

5.1 Use Cases

- Founder tokens that vest over time.
- Migrating existing ERC20 tokens and converting them into Petyl tokens.
- Converting base tokens into dividend tokens

5.2 Adding Converters

Main contract owner can assign new converters for any of its underlying tokens. Only the Main Contract summoner is only able to add token converters. This is to ensure the security of the system. Testing is available for basic checks on token converters to ensure a 1 to 1 converting of tokens is kept true.

5.3 Converting Tokens

CHAPTER 6

Token Rules

Base tokens have a smart contract hook which allows an arbitrary set of rules be attached to the movement of those tokens. There are a number of use cases where this is interesting.

6.1 Examples

Restrict project founders from selling tokens before a set vesting date Limit tokens to be traded between employees or investors on a whitelist Automated compliance for specific asset types

CHAPTER 7

Whitelists

Base tokens have a smart contract hook which allows an arbitrary set of rules be attached to the movement of those tokens. As part of the rules, tokens can call on a token whitelist and return a true or false.

CHAPTER 8

Documents

Petyl contracts are able to hold a list of document hashes for a variety of usecases.

The main usecase is to store founding contracts or documents relating to a project if required.

Notifications, announcements and disclosures are also covered in the publishing of documents

Documents can be stored directly onchain, IPFS or mapped to an external website address.

CHAPTER 9

Token Types

There are a number of token types, both base tokens and various combinations

- *Base tokens*
- *Dividend Tokens*
- Security Tokens
- *Employee Share Options*
- *Voting Tokens*

CHAPTER 10

Base Tokens

Base token type is the foundation of the Petyl protocol. It is essentially an ERC20 with some advanced features from the ERC777 standard and some bonus code to make them work with the Petyl contract.

10.1 Functions

Full list of functions can be found in the Reference docs.

10.2 Send vs Transfer

This is a naming convention from both ERC20 and ERC777 which is used to distinguish them. We have included both the Transferred and Sent events to comply with both standards.

10.3 Transfer with data

Certain use cases need additional data to be transferred along with token operations. We have included the `transferWithData()` function to support additional functionality both in sending and receiving tokens.

10.4 Operators

Token holders can assign an operator for repeated functionality. Any base token added to the Petyl contract will act as an operator for that token.

10.5 ERC1820 Registry

This is a key part of the ERC777 standard and used as a hook for additional functionality within the Petyl protocol.

CHAPTER 11

Distribution Tokens

Distribution tokens are base tokens with additional functions to both keep track and distribute tokens to holders. Due to the internal accounting required to keep track of who is owed what amount, dividend tokens must be minted separately to base tokens.

11.1 Functions

Full list of functions can be found in the Reference docs.

11.2 Deposit Dividends

Accepts up to 5 different ERC20 tokens as well as ETH Limited to a set number of ERC20 tokens in V1 due to gas constraints and security implications. Only the owner can add new ERC20 tokens to be accepted and paid out. Non approved tokens are not lock and can be transfered by the contract owner.

11.3 Withdraw Funds

Anyone can pull out the amount of tokens they have earned over the time of holding a dividend token. Transfers are precomputed with dividends owing paid out first before sending to another party.

The main component of the Petyl Protocol

12.1 Components

- Partitions
- Token Converters
- Documents

12.2 Partitions

Different tranches or classes of shares can be represented by separate tokens, mapped by the main Petyl contract. The

12.3 Token Converters

12.4 Documents

The main Petyl contract for each new venture comes built in with Documents

CHAPTER 13

Options

Time based conversions of tokens are made by creating a base token with rules and a token converter. The token rules can be as arbitrary or complex as written in Solidity. The user then calls the converter when these new options tokens meet the time based rules for transfer.

Examples include a tranche of founder shares which require 4 years vesting, which can be unlocked proportionally over time. Employee share offers are also possible with time based rules programmed in, with accounts checked off on a white list if conditions are met.

CHAPTER 14

Dutch Auctions

We believe that the market should determine the price of a token rather than you or I. The Dutch Auction mechanism achieves this fairness for two main reasons. First, at every auction everyone who buys a token, buys it at the same price. Second, the market determines that price.

14.1 How a Dutch Auction Works

Every Dutch auction has a set amount of tokens available to be bought. The price is initially set at a high price and during the auction continues to drop. The price drops until either all the tokens are sold, or hits a minimum price.

14.2 Set Up

Each dutch auction is defined by a start and end time, starting and minimum token price and the number of tokens to be sold. From this, the linear pricing is determined auction begins automatically. Users are invited to contribute funds as soon as the start time begins.

14.3 More information

This work was inspired by both the Algorand and Gnosis dutch auctions. More details can be found here: [Algo Auction](#) We do not support the refund policy but may do so in the future.

CHAPTER 15

Links

List of links related to the Petyl Protocol.

ERC1400 token standard