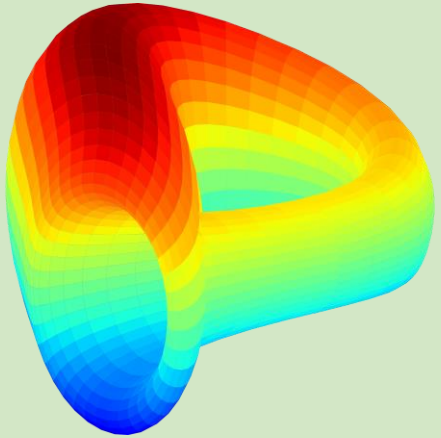
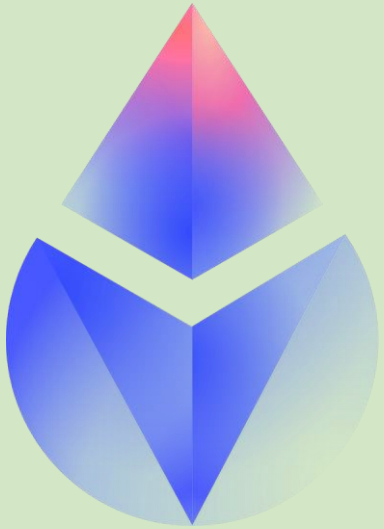
snekmate

Past | Present | Future

First and foremost, thank you for organising the Vyper Day 2023!





A few words about myself

Writing 🐍 Vyper code energises your body while it lets you sleep well due to built-in footgun protections.

Scan me!

https://github.com/pcaversaccio/snekmate

How everything started

sudo rm -rf --no-preserve-root /
@pcaversaccio

I think it would be fun to write vymate, the Vyper equivalent of solmate. Anyone already working on this? Otherwise I will probably start my own repo. And I will be super opinionated on any PR just to follow the solmate philosophy lmfaoooo

10:00 AM · Jul 30, 2022

Commit

Initial commit

main
v0.0.4 ... v0.0.1

pcaversaccio committed on Jul 30, 2022 Verified

Showing 1 changed file with 2 additions and 0 deletions.

2 README.md

```
@@ -0,0 +1,2 @@
+ # snekmate
+ State-of-the-art, highly opinionated, hyper-optimised, and secure Vyper smart contract building blocks.
```

0 comments on commit 70e4d4c

banteg @bantg · Jul 30, 2022
let's see if vyper need its own solmate

- auth: covered by the new Enum type
- tokens: covered by ape template cookiecutter plugin
- utils:
  - fixed point math: native feature
  - reentrancy guard: native feature
  - safe transfer: native feature

4 3 23

t11s
@transmissions11

> tokens: covered by ape template cookiecutter plugin

not everyone uses ape :p

> auth: covered by the new Enum type

not really? thats just one implementation detail

> fixed point math: native feature

but only for 10 decimal types? FPMLib is for wads

10:12 AM · Jul 30, 2022

Releases / v0.0.1

snekmate v0.0.1 🥳

pcaversaccio released this Mar 6 · 160 commits to main since this release · v0.0.1 · bc70a3b

Summary

I am just fucking proud and so happy to release the first official version of `snekmate` 🥳!

Honestly, what a feat:

- 582 commits
- 6,471 source lines of Vyper code
- 15,137 source lines of Solidity test code

🙏 Special thanks go to all the contributors who supported me during this journey, you're all badass!

- @jtriley-eth
- @YamenMerhi
- @jaglinux
- @bout3fiddy
- @hrik2001
- @mattsse
- @charles-cooper

Releases / v0.0.2

snekmate v0.0.2

pcaversaccio released this Jun 7 · 112 commits to main since this release · v0.0.2 · a4c5612

Summary

Vyper shipped version `0.3.9` nine days ago, I ship snekmate version `0.0.2` today targeting the latest Vyper version!

> Talk is cheap. Show me the code.

For all the math 🤓 nerds, I added `wad_ln` and `wad_exp` to the standard mathematical utility functions! Check them out here. Another noteworthy change is that I have added default support for EIP-5267 for all contracts that support EIP-712, i.e. `ERC20`, `ERC721`, and `ERC4626`.

Important: The default EVM version since Vyper version `0.3.8` is set to `shanghai` (i.e. the EVM includes the `PUSH0` instruction).

👇 Below you find the detailed code changes and the full CHANGELOG. On that note, have a great week and keep grinding anon!

Releases / v0.0.3

snekmate v0.0.3

pcaversaccio released this Oct 12 · 28 commits to main since this release · v0.0.3 · ebd8038

Summary

🐻 Bear markets are made for building and you know what, the sneeek is shipping today 🔥! You might have heard that Vyper shipped version `0.3.10` eight days ago. But you might not know that the latest Vyper version is a heavily performance-oriented version that, among other things, generates selector tables that now offer *O(1) performance* 🤯 (see PR #3496). And so that you can take advantage of the latest improvements, I ship snekmate version `0.0.3` today targeting the latest Vyper version!

Important: The default EVM version since Vyper version `0.3.8` is set to `shanghai` (i.e. the EVM includes the `PUSH0` instruction).

👇 Below you find the detailed code changes and the full `CHANGELOG`. Don't stop. Push harder. Keep going.

yearn / yearn-vaults-v3

Issues 3 | Pull requests | Discussions | Actions | Projects | Security | Insights

support eip-1271 permits #170

Open · banteg opened this issue on Jun 15 · 0 comments

banteg commented on Jun 15 • edited

https://eips.ethereum.org/EIPS/eip-1271

good implementation in snekmate
https://github.com/pcaversaccio/snekmate/blob/main/src/utils/SignatureChecker.vy

👍 1

snekmate 0.0.4

```
pip install snekmate
```

Install

```
> npm i snekmate
```

tricrypto-ng / contracts / main / CurveCryptoMathOptimized3.vy

Code | Blame | 878 lines (737 loc) · 29.3 KB

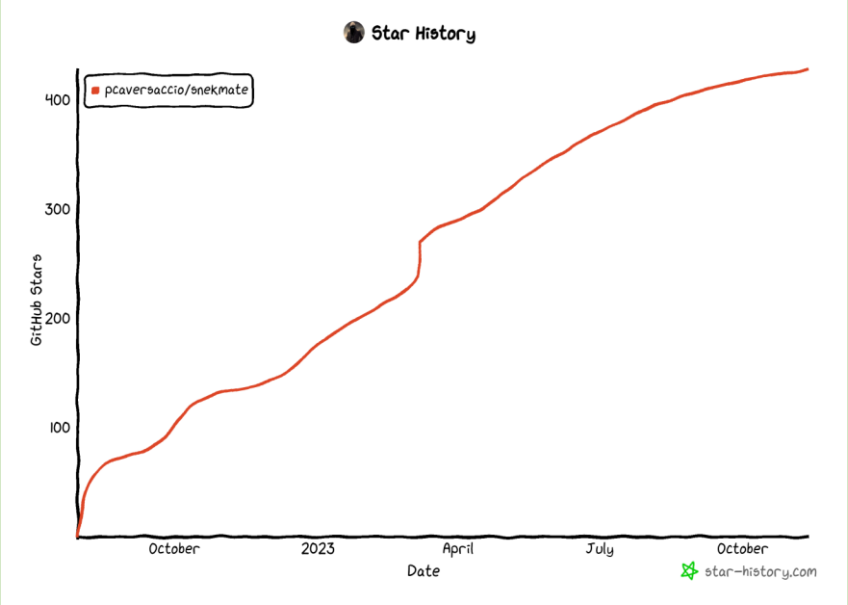
```
659         return K
660
661
662     @internal
663     @pure
664     def _snekmate_wad_exp(x: int256) -> uint256:
665
666         """
667         @dev Calculates the natural exponential function of a signed integer with
668              a precision of 1e18.
669         @notice Note that this function consumes about 810 gas units. The implementation
670                 is inspired by Remco Bloemen's implementation under the MIT license here:
671                 https://xn--2-umb.com/22/exp-ln.
672         @dev This implementation is derived from Snekmate, which is authored
673              by pcaversaccio (Snekmate), distributed under the AGPL-3.0 license.
674              https://github.com/pcaversaccio/snekmate
675         @param x The 32-byte variable.
676         @return int256 The 32-byte calculation result.
677         """
678         value: int256 = x
```

What has happened since then

742 Commits

Contributors 13

+ 2 contributors



```
src
├── auth
│   ├── Ownable — "Owner-Based Access Control Functions"
│   ├── Ownable2Step — "2-Step Ownership Transfer Functions"
│   ├── AccessControl — "Multi-Role-Based Access Control Functions"
│   └── interfaces
│       └── IAccessControl — "AccessControl Interface Definition"
├── extensions
│   ├── ERC2981 — "ERC-721 and ERC-1155 Compatible ERC-2981 Reference Implementation"
│   ├── ERC4626 — "Modern and Gas-Efficient ERC-4626 Tokenised Vault Implementation"
│   └── interfaces
│       └── IERC2981 — "EIP-2981 Interface Definition"
├── tokens
│   ├── ERC20 — "Modern and Gas-Efficient ERC-20 + EIP-2612 Implementation"
│   ├── ERC721 — "Modern and Gas-Efficient ERC-721 + EIP-4494 Implementation"
│   ├── ERC1155 — "Modern and Gas-Efficient ERC-1155 Implementation"
│   └── interfaces
│       ├── IERC20Permit — "EIP-2612 Interface Definition"
│       ├── IERC721Enumerable — "EIP-721 Optional Enumeration Interface Definition"
│       ├── IERC721Metadata — "EIP-721 Optional Metadata Interface Definition"
│       ├── IERC721Permit — "EIP-4494 Interface Definition"
│       ├── IERC721Receiver — "EIP-721 Token Receiver Interface Definition"
│       ├── IERC1155 — "EIP-1155 Interface Definition"
│       ├── IERC1155MetadataURI — "EIP-1155 Optional Metadata Interface Definition"
│       ├── IERC1155Receiver — "EIP-1155 Token Receiver Interface Definition"
│       └── IERC4906 — "EIP-4906 Interface Definition"
└── utils
    ├── Base64 — "Base64 Encoding and Decoding Functions"
    ├── BatchDistributor — "Batch Sending Both Native and ERC-20 Tokens"
    ├── CreateAddress — "`CREATE` EVM Opcode Utility Function for Address Calculation"
    ├── Create2Address — "`CREATE2` EVM Opcode Utility Functions for Address Calculations"
    ├── ECDSA — "Elliptic Curve Digital Signature Algorithm (ECDSA) Functions"
    ├── SignatureChecker — "ECDSA and EIP-1271 Signature Verification Functions"
    ├── EIP712DomainSeparator — "EIP-712 Domain Separator"
    ├── Math — "Standard Mathematical Utility Functions"
    ├── MerkleProofVerification — "Merkle Tree Proof Verification Functions"
    ├── Multicall — "Multicall Functions"
    └── interfaces
        ├── IERC5267 — "EIP-5267 Interface Definition"
```

So, what on earth is snekmate?

- GitHub: https://github.com/pcaversaccio/snekmate
- State-of-the-art, highly opinionated, hyper-optimised, and secure Vyper smart contract building blocks.

snekmate's design principles

- Security first
- Maximum composability
- Don't build for yourself, but for what is/will be used
- All contracts should be maximally human-readable

fiddy
@fiddyresearch

Snekmate has been very crucial in making new curve amm contracts safu with their mul_div. And it will be an important tool when vyper modules gets launched.

sudo rm -rf --no-preserve-root / @pcaversaccio · Mar 3

anon, we all know you love cube roots, but you were checking Solmate & Solady and couldn't find a function. Check snekmate now, where u can calculate the cbrt & wad cbrt for under 2,000 gas! Credit goes to @_bout3fiddy_, @newmichwill & @CurveFinance!
github.com/pcaversaccio/s...

```
l) -> uint256:

 of an unsigned integer.
on consumes about 1,950 to 2,050 gas u
e of `x` and `roundup`. The implementa
ance's implementation under the MIT li
rvefi/tricrypto-ng/blob/main/contracts
 from which the cube root is calculate
riable that specifies whether
e default `False` is round down.
 of `x`.
```

```
:

f an unsigned integer with a preci

 consumes about 1,850 to 1,950 gas
of `x`. The implementation is insp
ementation under the MIT license b
efi/tricrypto-ng/blob/main/contrac
rom which the cube root is calcula
of `x` with a precision of 1e18.
```

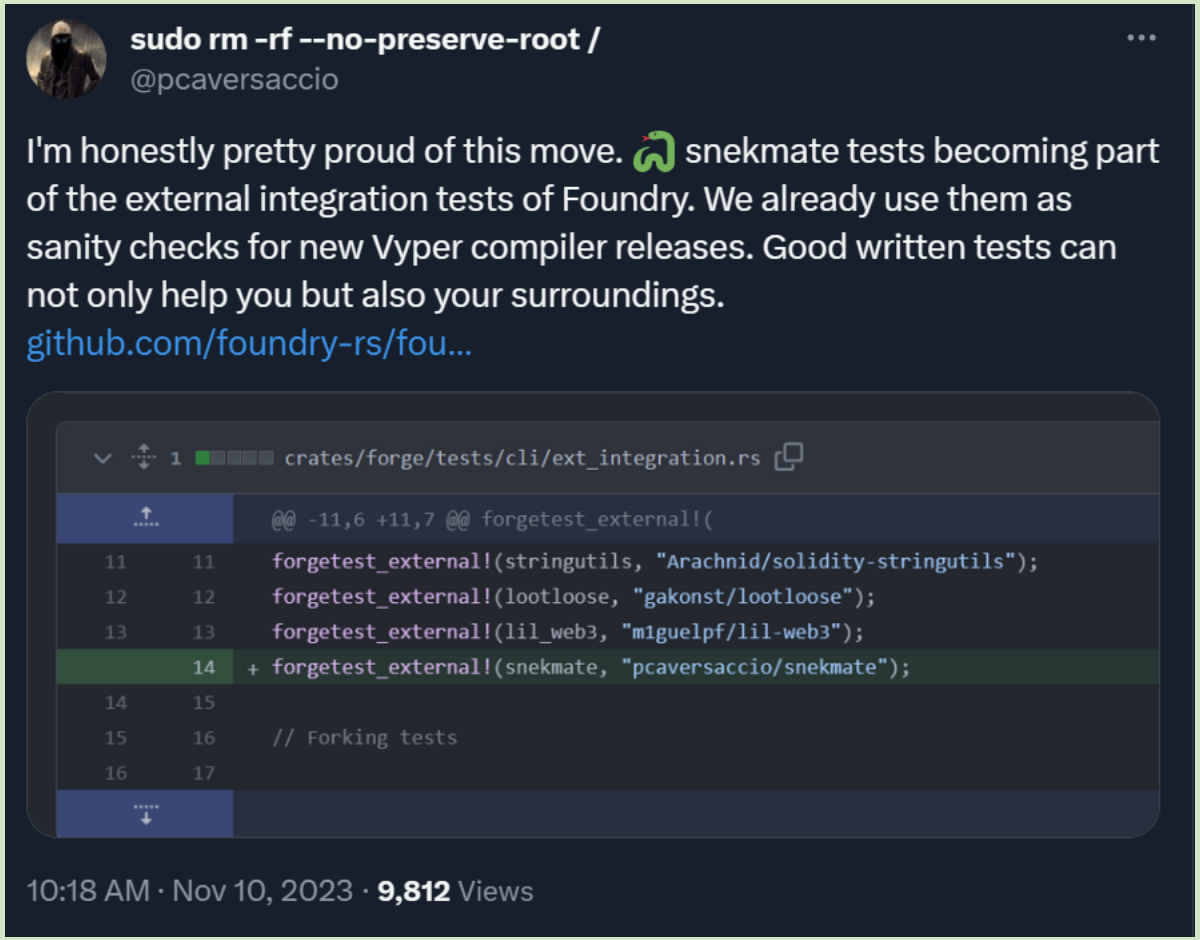
3:15 PM · Mar 3, 2023 · **3,625** Views

Two example projects that use snekmate contracts in production

- Curve
- Cog Finance

```
# @version 0.3.9

"""
@title Fuse Box
@author cog.finance
@license AGPL-3.0
@notice A robust Oracle Implementation with secure upgradability in mind, and simplicity at its core.
"""

# ////////////////////////////////////////////////////// #
#               2Step Ownership Ty Snekmate              #
# ////////////////////////////////////////////////////// #

# @dev Returns the address of the current owner.
owner: public(address)


# @dev Returns the address of the pending owner.
pending_owner: public(address)
```

Stateless & stateful fuzz the shit out of everything



Tests

This repository contains Foundry-based unit tests, property-based tests (i.e. fuzzing), and invariant tests for all contracts, if applicable. All tests are run as part of the CI pipeline `test-contracts`.

> **Note**: An *invariant* is a property of a program that should always hold true. Fuzzing is a way of checking whether the invariant is falsifiable.

Contract	Unit Tests	Property-Based Tests	Invariant Tests
`Ownable`	✅	✅	✅
`Ownable2Step`	✅	✅	✅
`AccessControl`	✅	✅	✅
`ERC2981`	✅	✅	✅
`ERC4626`	✅	✅	✅
`ERC20`	✅	✅	✅
`ERC721`	✅	✅	✅
`ERC1155`	✅	✅	✅
`Base64`	✅	❌	❌
`BatchDistributor`	✅	✅	✅
`CreateAddress`	✅	✅	❌
`Create2Address`	✅	✅	❌
`ECDSA`	✅	✅	❌
`SignatureChecker`	✅	✅	❌
`EIP712DomainSeparator`	✅	✅	❌
`Math`	✅	✅	❌
`MerkleProofVerification`	✅	✅	❌
`Multicall`	✅	❌	❌

✅ Test Type Implemented ❌ Test Type Not Implemented

ok i need MODULES TO GO UP!

sudo init vyper
@big_tech_sux

ok i need MODULES TO GO UP. every day i am checking vyperlang github, no modules commits. every day, check github, see create2 commits, abi_decode commits, enum commits. NO MODULE COMMITS. i can't take this anymore. i need modules to GO UP ALREADY. can devs DO SOMETHING.

10:45 PM · Jun 9, 2022

sudo rm -rf --no-preserve-root /
@pcaversaccio

ok i need MODULES TO GO UP. every day i am checking vyperlang github, no module commits... **rechecking repo** ooooh wait THERE ARE MODULE COMMITS 🤔. i can't take this anymore. i need modules to GO UP ALREADY NOW. OH wait, @big_tech_sux is DOING SOMETHING NOW.

sudo init vyper @big_tech_sux · Nov 7
🫡

```
(vyper) ~/vyper $ cat tmp/libuser.vy
# foo.vy
import lib as lib

@external
def foo() -> int256:
    return lib.foo()

(vyper) ~/vyper $ cat tmp/lib.vy
# lib.vy

@internal
def bar() -> int256:
    return 1

@internal
def foo() -> int256:
    return self.bar()

(vyper) ~/vyper $ vyc tmp/libuser.vy
0x61004561000f6000396100456000f35f3560e01c63c298557881186
100245734610041576020610020606061002f565b6060f35b5f5ffd5b
6001815250565b610039604061002856 5b604051815250565b5f80fd8
418458000a165767970657283000 30b0012
```

9:27 AM · Nov 7, 2023 · **2,451** Views

Modules will unleash the full potential

- This is an illustration of how snekmate code snippets may be used in the future, but the modules design specification is not yet finalised and may therefore still change significantly!

```
# pragma version ^0.4.0

from snekmate import ECDSA as ecdsa
from snekmate import EIP712DomainSeparator as eip712_domain_separator

...

@external
def permit(owner: address, spender: address, amount: uint256, deadline: uint256, v: uint8, r: bytes32, s: bytes32):
    assert block.timestamp <= deadline, "ERC20Permit: expired deadline"

    current_nonce: uint256 = self.nonces[owner]
    self.nonces[owner] = unsafe_add(current_nonce, 1)

    struct_hash: bytes32 = keccak256(_abi_encode(_PERMIT_TYPE_HASH, owner, spender, amount, current_nonce, deadline))
    hash: bytes32  = eip712_domain_separator._hash_typed_data_v4(struct_hash)

    signer: address = ecdsa._recover_vrs(hash, convert(v, uint256), convert(r, uint256), convert(s, uint256))
    assert signer == owner, "ERC20Permit: invalid signature"

    self._approve(owner, spender, amount)
```

Roadmap

- [ ] Integrate snekmate Contracts with Halmos — feature, help wanted
  #180 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] Add Slither to CI Pipeline — ci/cd, feature
  #179 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] `TimelockController` Contract — feature
  #178 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] `Governor` Contract — feature
  #177 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] Vyper-Based Multisig Wallet — feature, help wanted
  #176 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] Add `P256Verifier` to snekmate — feature, optimisation
  #175 opened 4 days ago by pcaversaccio — 0.0.5
- [ ] `ERC-4337` Functionalities — feature, help wanted
  #174 opened 4 days ago by pcaversaccio — 5 tasks — 0.0.5

`ERC-4337` Functionalities #174

Open — 5 tasks — pcaversaccio opened this issue 4 days ago · 0 comments

pcaversaccio commented 4 days ago • edited — Owner

I consider the following ERC-4337 functionalities important to have for snekmate:

- [ ] `SimpleAccount` (example); additionally to the common `secp256k1`-based elliptic curve verification, we should also offer a variant with the `secp256r1`-based elliptic curve verification. I have written a Vyper-based verifier here; also see Add `P256Verifier` to snekmate #175;
- [ ] `SimpleAccountFactory` (example);
- [ ] `BLSAccount` (example); minimal BLS-based account that uses an aggregated signature;
- [ ] `BLSAccountFactory` (example);
- [ ] `TokenPaymaster` (example); a sample `ERC-20` token paymaster for `ERC-4337`;

References

- https://github.com/eth-infinitism/account-abstraction
- https://github.com/de33/vyper-erc4337

Vyper-Based Multisig Wallet #176

Open — pcaversaccio opened this issue 4 days ago · 1 comment

pcaversaccio commented 4 days ago — Owner

It's time for Vyper to catch up with Solidity on the multisig front, and snekmate plans to offer its own Vyper-based multisig wallet that is capable of reaching mainstream adoption.

References

- https://github.com/dmfxyz/supersig
- https://github.com/ricobank/multisig

Q & A