# Developer Tools

A guide to available tools, components, patterns, and platforms for developing applications on EVM-compatible chains.

## New developers start here

- [Solidity](#)
-
    - The most popular smart contract language.
- [Metamask](#)
-
    - Browser extension wallet to interact with Dapps.
- [thirdweb](#)
-
    - SDKs in every language, smart contracts, tools, and infrastructure for web3 development.
- [Truffle](#)
-
    - Most popular smart contract development, testing, and deployment framework. Install the cli via npm and start here to write your first smart contracts.
- [Truffle boxes](#)
-
    - Packaged components for the Ethereum ecosystem.
- [Hardhat](#)
-
    - Flexible, extensible and fast Ethereum development environment.
- [Cryptotux](#)
-
    - A Linux image ready to be imported in VirtualBox that includes the development tools mentionned above
- [OpenZeppelin Starter Kits](#)
-
    - An all-in-one starter box for developers to jumpstart their smart contract backed applications. Includes Truffle, OpenZeppelin SDK, the OpenZeppelin/contracts-ethereum-package EVM package of audited smart contract, a react-app and rimble for easy styling.
- [EthHub.io](#)
-
    - Comprehensive crowdsourced overview of Ethereum- its history, governance, future plans and development resources.
- [EthereumDev.io](#)
-
    - The definitive guide for getting started with Ethereum smart contract programming.
- [Brownie](#)
-
    - Brownie is a Python framework for deploying, testing and interacting with Ethereum smart contracts.
- [Ethereum Stack Exchange](#)
-
    - Post and search questions to help your development life cycle.
- [dfuse](#)
-
    - Slick blockchain APIs to build world-class applications.
- [Biconomy](#)
-
    - Do gasless transactions in your dapp by enabling meta-transactions using simple to use SDK.
- [Blocknative](#)
- — Blockchain events before they happen. Blocknative's portfolio of developers tools make it easy to build with mempool data.
- [useWeb3.xyz](#)
- — A curated overview of the best and latest resources on Ethereum, blockchain and Web3 development.

## Developing Smart Contracts

### Smart Contract Languages

- [Solidity](#)
-
    - Ethereum smart contracting language
- [Vyper](#)

- New experimental pythonic programming language

## Frameworks

- [thirdweb](#)
  - Provides the tools needed to build custom smart contracts efficiently by offering a set of prebuilt base contracts and a set of reusable components, or extensions, that can be integrated into your own smart contracts.
- [Truffle](#)
  - Most popular smart contract development, testing, and deployment framework. The Truffle suite includes Truffle,[Ganache](#)
- , and[Drizzle](#)
- .[Deep dive on Truffle here](#)
- [Hardhat](#)
  - Flexible, extensible and fast Ethereum development environment.
- [Brownie](#)
  - Brownie is a Python framework for deploying, testing and interacting with Ethereum smart contracts.
- [Embark](#)
  - Framework for DApp development
- [Waffle](#)
  - Framework for advanced smart contract development and testing, small, flexible, fast (based on ethers.js)
- [Dapp](#)
  - Framework for DApp development, successor to DApple
- [Etherlime](#)
  - ethers.js based framework for Dapp deployment
- [Parasol](#)
  - Agile smart contract development environment with testing, INFURA deployment, automatic contract documentation and more. It features a flexible and unopinionated design with unlimited customizability
- [0xcert](#)
  - JavaScript framework for building decentralized applications
- [OpenZeppelin SDK](#)
  - OpenZeppelin SDK: A suite of tools to help you develop, compile, upgrade, deploy and interact with smart contracts.
- [sbt-ethereum](#)
  - A tab-completey, text-based console for smart-contract interaction and development, including wallet and ABI management, ENS support, and advanced Scala integration.
- [Cobra](#)
  - A fast, flexible and simple development environment framework for Ethereum smart contract, testing and deployment on Ethereum virtual machine(EVM).
- [Epirus](#)
  - Java framework for building smart contracts.

## IDEs

- [Remix](#)
  - Web IDE with built in static analysis, test blockchain VM.
- [Ethereum Studio](#)
  - Web IDE. Built in browser blockchain VM, Metamask integration (one click deployments to Testnet/Mainnet), transaction logger and live code your WebApp among many other features.
- [Atom](#)
  - Atom editor with[Atom Solidity Linter](#)

- ,[Etheratom](#)
- ,[autocomplete-solidity](#)
- , and[language-solidity](#)
- packages
- [Vim solidity](#)
-
    - Vim syntax file for solidity
- [Visual Studio Code](#)
-
    - Visual Studio Code extension that adds support for Solidity
- [Ethcode](#)
-
    - Visual Studio Code extension to compile, execute & debug Solidity & Vyper programs
- [Intellij Solidity Plugin](#)
-
    - Open-source plug-in for[JetBrains IntelliJ Idea IDE](#)
- (free/commercial) with syntax highlighting, formatting, code completion etc.
- [YAKINDU Solidity Tools](#)
-
    - Eclipse based IDE. Features context sensitive code completion and help, code navigation, syntax coloring, build in compiler, quick fixes and templates.
- [Eth Fiddle](#)
-
    - IDE developed by[The Loom Network](#)
- that allows you to write, compile and debug your smart contract. Easy to share and find code snippets.

## Other tools

- [Atra Blockchain Services](#)
-
    - Atra provides web services to help you build, deploy, and maintain decentralized applications on the Ethereum blockchain.
- [Azure Blockchain Dev Kit for Ethereum for VSCode](#)
-
    - VSCode extension that allows for creating smart contracts and deploying them inside of Visual Studio Code

## Test Blockchain Networks

- [ethnode](#)
-
    - Run an Ethereum node (Geth or Parity) for development, as easy as npm i -g ethnode && ethnode
- .
- [Ganache](#)
-
    - App for test Ethereum blockchain with visual UI and logs
- [Kaleido](#)
-
    - Use Kaleido for spinning up a consortium blockchain network. Great for PoCs and testing
- [Besu Private Network](#)
-
    - Run a private network of Besu nodes in a Docker container
- **[Orion](#)
-
    - Component for performing private transactions by PegaSys
- **[Artemis](#)
-
    - Java implementation of the Ethereum 2.0 Beacon Chain by PegaSys
- [Cliquebait](#)
-
    - Simplifies integration and accepting testing of smart contract applications with docker instances that closely resembles a real blockchain network
- [Local Raiden](#)
-
    - Run a local Raiden network in docker containers for demo and testing purposes
- [Private networks deployment scripts](#)
-
    - Out-of-the-box deployment scripts for private PoA networks

- [Local Ethereum Network](#)
  - Out-of-the-box deployment scripts for private PoW networks
- [Ethereum on Azure](#)
  - Deployment and governance of consortium Ethereum PoA networks
- [Ethereum on Google Cloud](#)
  - Build Ethereum network based on Proof of Work
- [Infura](#)
  - Ethereum API access to Ethereum networks (Mainnet, Ropsten, Rinkeby, Goerli, Kovan)
- [CloudFlare Distributed Web Gateway](#)
  - Provides access to the Ethereum network through the Cloudflare instead of running your own node
- [Chainstack](#)
  - Shared and dedicated Ethereum nodes as a service (Mainnet, Ropsten)
- [Alchemy](#)
  - Blockchain Developer Platform, Ethereum API, and Node Service (Mainnet, Ropsten, Rinkeby, Goerli, Kovan)
- [ZMOK](#)
  - JSON-RPC Ethereum API (Mainnet, Rinkeby, Front-running Mainnet)
- [Watchdata](#)
  - Provide simple and reliable API access to Ethereum blockchain

## Alternative Celo Faucet

- [LearnWeb3 Faucet](#)

## Test Ether Faucets

- [Rinkeby faucet](#)
- [Kovan faucet](#)
- [Ropsten faucet (MetaMask)](#)
- [Ropsten faucet (rpanic)](#)
- [Goerli faucet](#)
- [Universal faucet](#)
- [Nethereum.Faucet](#)
  - A C#/.NET faucet

# Communicating with Ethereum

## Frontend Ethereum APIs

- [thirdweb](#)
  - Build web3 applications that can interact with your smart contracts using our powerful SDKs.
- [Web3.js](#)
  - Javascript Web3
- [Eth.js](#)
  - Javascript Web3 alternative
- [Ethers.js](#)
  - Javascript Web3 alternative, useful utilities and wallet features
- [useDApp](#)
  - React based framework for rapid DApp development on Ethereum
- [light.js](#)
- A high-level reactive JS library optimized for light clients.
- [Web3Wrapper](#)
  - Typescript Web3 alternative

- [Ethereumjs](#)
-
    - A collection of utility functions for Ethereum like [ethereumjs-util](#)
- and [ethereumjs-tx](#)
- [Alchemy-web3.js](#)
-
    - Javascript Web3 wrapper with automatic retries, access to [Alchemy's enhanced APIs](#)
- , and robust websocket connections.
- [flex-contract](#)
- and [flex-ether](#)
-
    - Modern, zero-configuration, high-level libraries for interacting with smart contracts and making transactions.
- [ez-ens](#)
-
    - Simple, zero-configuration Ethereum Name Service address resolver.
- [web3x](#)
-
    - A TypeScript port of web3.js. Benefits includes tiny builds and full type safety, including when interacting with contracts.
- [Nethereum](#)
-
    - Cross-platform Ethereum development framework
- [dfuse](#)
-
    - A TypeScript library to use [dfuse Ethereum API](#)
- [Drizzle](#)
-
    - Redux library to connect a frontend to a blockchain
- [Tasit SDK](#)
-
    - A JavaScript SDK for making native mobile Ethereum dapps using React Native
- [useMetamask](#)
-
    - a custom React Hook to manage Metamask in Ethereum ÐApp projects
- [WalletConnect](#)
-
    - Open protocol for connecting Wallets to Dapps
- [Subproviders](#)
-
    - Several useful subproviders to use in conjunction with [Web3-provider-engine](#)
- (including a LedgerSubprovider for adding Ledger hardware wallet support to your dApp)
- Strictly Typed - Javascript alternatives* [elm-ethereum](#)
-
    - [purescript-web3](#)
- [ChainAbstractionLayer](#)
-
    - Communicate with different blockchains (including Ethereum) using a single interface.
- [Delphereum](#)
-
    - a Delphi interface to the Ethereum blockchain that allows for development of native dApps for Windows, macOS, iOS, and Android.
- [Torus](#)
-
    - Open-sourced SDK to build dapps with a seamless onboarding UX
- [Fortmatic](#)
-
    - A simple to use SDK to build web3 dApps without extensions or downloads.
- [Portis](#)
-
    - A non-custodial wallet with an SDK that enables easy interaction with DApps without installing anything.
- [create-eth-app](#)
-
    - Create Ethereum-powered front-end apps with one command.
- [Scaffold-ETH](#)
-
    - Beginner friendly forkable github for getting started building smart contracts.
- [Notify.js](#)
-

- Deliver real-time notifications to your users. With built-in support for Speed-Ups and Cancels, Blocknative Notify.js helps users transact with confidence. Notify.js is easy to integrate and quick to customize.

## Backend Ethereum APIs

- [thirdweb](#)
  - Build web3 applications that can interact with your smart contracts using our powerful SDKs.
- [Web3.py](#)
  - Python Web3
- [Web3.php](#)
  - PHP Web3
- [Ethereum-php](#)
  - PHP Web3
- [Web3j](#)
  - Java Web3
- [Nethereum](#)
  - .Net Web3
- [Ethereum.rb](#)
  - Ruby Web3
- [rust-web3](#)
  - Rust Web3
- [Web3.hs](#)
  - Haskell Web3
- [KEthereum](#)
  - Kotlin Web3
- [Eventeum](#)
  - A bridge between Ethereum smart contract events and backend microservices, written in Java by Kauri
- [Ethereumex](#)
  - Elixir JSON-RPC client for the Ethereum blockchain
- [Ethereum-jsonrpc-gateway](#)
  - A gateway that allows you to run multiple Ethereum nodes for redundancy and load-balancing purposes. Can be ran as an alternative to (or on top of) Infura. Written in Golang.
- [EthContract](#)
  - A set of helper methods to help query ETH smart contracts in Elixir
- [Ethereum Contract Service](#)
  - A MESG Service to interact with any Ethereum contract based on its address and ABI.
- [Ethereum Service](#)
  - A MESG Service to interact with events from Ethereum and interact with it.
- [Marmo](#)
  - Python, JS, and Java SDK for simplifying interactions with Ethereum. Uses relayers to offload transaction costs to relayers.
- [Ethereum Logging Framework](#)
  - provides advanced logging capabilities for Ethereum applications and networks including a query language, query processor, and logging code generation
- [Watchdata](#)
  - Provide simple and reliable API access to Ethereum blockchain

## Bootstrap/Out-of-Box tools

- [Truffle boxes](#)
  - Packaged components for the Ethereum ecosystem
- [Create Eth App](#)
  - Create Ethereum-powered frontend apps with one command
- [Besu Private Network](#)
  - Run a private network of Besu nodes in a Docker container
- [Testchains](#)
  - Pre-configured .NET devchains for fast response (PoA)
- \*\*[Blazor/Blockchain Explorer](#)
  - Wasm blockchain explorer (functional sample)
- [Local Raiden](#)
  - Run a local Raiden network in docker containers for demo and testing purposes
- [Private networks deployment scripts](#)
  - Out-of-the-box deployment scripts for private PoA networks
- [Parity Demo-PoA Tutorial](#)
  - Step-by-Step tutorial for building a PoA test chain with 2 nodes with Parity authority round consensus
- [Local Ethereum Network](#)
  - Out-of-the-box deployment scripts for private PoW networks
- [Kaleido](#)
  - Use Kaleido for spinning up a consortium blockchain network. Great for PoCs and testing
- [Cheshire](#)
  - A local sandbox implementation of the CryptoKitties API and smart contracts, available as a Truffle Box
- [aragonCLI](#)
  - aragonCLI is used to create and develop Aragon apps and organizations.
- [ColonyJS](#)
  - JavaScript client that provides an API for interacting with the Colony Network smart contracts.
- [ArcJS](#)
  - Library that facilitates javascript application access to the DAOstack Arc ethereum smart contracts.
- [Arkane Connect](#)
  - JavaScript client that provides an API for interacting with Arkane Network, a wallet provider for building user-friendly dapps.
- [Onboard.js](#)
  - Blocknative Onboard is the quick and easy way to add multi-wallet support to your project. With built-in modules for more than 20 unique hardware and software wallets, Onboard saves you time and headaches.
- [web3-react](#)
  - React framework for building single-page Ethereum dApps

## Ethereum ABI (Application Binary Interface) tools

- [Online ABI encoder](#)
  - Free ABI encoder online service that allows you to encode your Solidity contract's functions and constructor arguments.
- [ABI decoder](#)
  - library for decoding data params and events from Ethereum transactions
- [ABI-gen](#)
  - Generate Typescript contract wrappers from contract ABI's.
- [Ethereum ABI UI](#)

- Auto-generate UI form field definitions and associated validators from an Ethereum contract ABI
- [headlong](#)
  - type-safe Contract ABI and Recursive Length Prefix library in Java
- [EasyDapper](#)
  - Generate dapps from Truffle artifacts, deploy contracts on public/private networks, offers live customizable public page to interact with contracts.
- [One Click dApp](#)
  - Instantly create a dApp at a unique URL using the ABI.
- [Truffle Pig](#)
  - a development tool that provides a simple HTTP API to find and read from Truffle-generated contract files, for use during local development. Serves fresh contract ABIs over http.
- [Ethereum Contract Service](#)
  - A MESG Service to interact with any Ethereum contract based on its address and ABI.
- [Nethereum-CodeGenerator](#)
  - A web based generator which creates a Nethereum based C# Interface and Service based on Solidity Smart Contracts.
- [EVMConnector](#)
  - Create shareable contract dashboards and interact with arbitrary EVM-based blockchain functions, with or without an ABI.

## Patterns & Best Practices

### Patterns for Smart Contract Development

- [Dappsys: Safe, simple, and flexible Ethereum contract building blocks](#)
  - has solutions for common problems in Ethereum/Solidity, eg.*[Whitelisting](#)
    - 
      - [Upgradable ERC20-Token](#)
    - 
      - [ERC20-Token-Vault](#)
    - 
      - [Authentication (RBAC)](#)
    - 
      - [...several more...](#)
  - provides building blocks for the[MakerDAO](#)
  - or[The TAO](#)
  - should be consulted before creating own, untested, solutions
  - usage is described in[Dapp-a-day 1-10](#)
  - and[Dapp-a-day 11-25](#)
- [OpenZeppelin Contracts: An open framework of reusable and secure smart contracts in the Solidity language](#).
  - Likely the most widely-used libraries and smart contracts
  - Similar to Dappsys, more integrated into Truffle framework
  - [Blog about Best Practices with Security Audits](#)
- [Advanced Workshop with Assembly](#)
- [Simpler Ethereum Multisig](#)
  - especially sectionBenefits
- [CryptoFin Solidity Auditing Checklist](#)

- A checklist of common findings, and issues to watch out for when auditing a contract for a mainnet launch.
- [aragonOS: A smart contract framework for building DAOs, Dapps and protocols](#)
  - Upgradeability: Smart contracts can be upgraded to a newer version
  - Permission control: By using theauth
  - andauthP
  - modifiers, you can protect functionality so only other apps or entities can access it
  - Forwarders: aragonOS apps can send their intent to perform an action to other apps, so that intent is forwarded if a set of requirements are met
- [EIP-2535 Diamond Standard](#)
  - Organize contracts so they share the same contract storage and Ethereum address.
  - Solves the 24KB max contract size limit.
  - Upgrade diamonds by adding/replacing/removing any number of functions in a single transaction.
  - Upgrades are transparent by recording them with a standard event.
  - Get information about a diamond with events and/or four standard functions.
- [Clean Contracts - A guide to writing clean code](#)

## Upgradeability

- [Blog von Elena Dimitrova, Dev at colony.io](#)
  - [https://blog.colony.io/writing-more-robust-smart-contracts-99ad0a11e948](https://blog.colony.io/writing-more-robust-smart-contracts-99ad0a11e948)
  - [https://blog.colony.io/writing-upgradeable-contracts-in-solidity-6743f0eecc88](https://blog.colony.io/writing-upgradeable-contracts-in-solidity-6743f0eecc88)
- [Aragon research blog](#)
  - [Library driven development](#)
  - [Advanced Solidity code deployment techniques](#)
- [OpenZeppelin on Proxy Libraries](#)

# Infrastructure

## Ethereum Clients

- [Besu](#)
  - an open-source Ethereum client developed under the Apache 2.0 license and written in Java. The project is hosted by Hyperledger.
- [Geth](#)
  - Go client
- [OpenEthereum](#)
  - Rust client, formerly called Parity
- [Aleth](#)
  - C++ client
- [Nethermind](#)
  - .NET Core client
- [Infura](#)
  - A managed service providing Ethereum client standards-compliant APIs
- [Trinity](#)
  - Python client using[py-evm](#)
- [Ethereumjs](#)

- 
  - JS client using[ethereumjs-vm](#)
- [Seth](#)
- 
  - Seth is an Ethereum client tool—like a "MetaMask for the command line"
- [Mustekala](#)
- 
  - Ethereum Light Client project of Metamask
- [Exthereum](#)
- 
  - Elixir client
- [EWF Parity](#)
- 
  - Energy Web Foundation client for the Tobalaba test network
- [Quorum](#)
- 
  - A permissioned implementation of Ethereum supporting data privacy by[JP Morgan](#)
- [Mana](#)
- 
  - Ethereum full node implementation written in Elixir.
- [Chainstack](#)
- 
  - A managed service providing shared and dedicated Geth nodes
- [QuickNode](#)
- 
  - Blockchain developer cloud with API access and node-as-a-service.
- [Watchdata](#)
- 
  - Provide simple and reliable API access to Ethereum blockchain

## Storage

- [IPFS](#)
- 
  - Decentralised storage and file referencing* [Mahuta](#)
- 
  - 
    - IPFS Storage service with added search capability, formerly IPFS-Store
- 
  - [OrbitDB](#)
- 
  - 
    - Decentralised database on top of IPFS
- 
  - [JS IPFS API](#)
- 
  - 
    - A client library for the IPFS HTTP API, implemented in JavaScript
- 
  - [TEMPORAL](#)
- 
  - 
    - Easy to use API into IPFS and other distributed/decentralised storage protocols
- 
  - [PINATA](#)
- 
  - 
    - The Easiest Way to Use IPFS
- [Swarm](#)
- 
  - Distributed storage platform and content distribution service, a native base layer service of the Ethereum web3 stack
- [Infura](#)
- 
  - A managed IPFS API Gateway and pinning service
- [3Box Storage](#)
- 
  - An api for user controlled, distrubuted storage. Built on top of IPFS and Orbitdb.

- [Aleph.im](#)
  - an offchain incentivized peer-to-peer cloud project (database, file storage, computing and DID) compatible with Ethereum and IPFS.

## Messaging

- [Whisper](#)
  - Communication protocol for DApps to communicate with each other, a native base layer service of the Ethereum web3 stack
- [DEVp2p Wire Protocol](#)
  - Peer-to-peer communications between nodes running Ethereum/Whisper
- [Pydevp2p](#)
  - Python implementation of the RLPx network layer
- [3Box Threads](#)
  - API to allow developers to implement IPFS persisted, or in memory peer to peer messaging.

# Testing Tools

- [Truffle Teams](#)
  - Zero-Config continuous integration for truffle projects
- [Solidity code coverage](#)
  - Solidity code coverage tool
- [Solidity coverage](#)
  - Alternative code coverage for Solidity smart-contracts
- [Solidity function profiler](#)
  - Solidity contract function profiler
- [Sol-profiler](#)
  - Alternative and updated Solidity smart contract profiler
- [Espresso](#)
  - Speedy, parallelised, hot-reloading solidity test framework
- [Eth tester](#)
  - Tool suite for testing Ethereum applications
- [Cliquebait](#)
  - Simplifies integration and accepting testing of smart contract applications with docker instances that closely resembles a real blockchain network
- [Hevm](#)
  - The hevm project is an implementation of the Ethereum virtual machine (EVM) made specifically for unit testing and debugging smart contracts
- [Ethereum graph debugger](#)
  - Solidity graphical debugger
- [Tenderly CLI](#)
  - Speed up your development with human readable stack traces
- [Solhint](#)
  - Solidity linter that provides security, style guide and best practice rules for smart contract validation
- [Ethlint](#)
  - Linter to identify and fix style & security issues in Solidity, formerly Solium
- [Decode](#)
  - npm package which parses tx's submitted to a local testrpc node to make them more readable and easier to understand

- [truffle-assertions](#)
  - An npm package with additional assertions and utilities used in testing Solidity smart contracts with truffle. Most importantly, it adds the ability to assert whether specific events have (not) been emitted.
- [Psol](#)
  - Solidity lexical preprocessor with mustache.js-style syntax, macros, conditional compilation and automatic remote dependency inclusion.
- [solpp](#)
  - Solidity preprocessor and flattener with a comprehensive directive and expression language, high precision math, and many useful helper functions.
- [Decode and Publish](#)
- – Decode and publish raw ethereum tx. Similar to [https://live.blockcypher.com/btc-testnet/decodetx/](https://live.blockcypher.com/btc-testnet/decodetx/)
- [Doppelgänger](#)
  - a library for mocking smart contract dependencies during unit testing.
- [rocketh](#)
  - A simple lib to test ethereum smart contract that allow to use whatever web3 lib and test runner you choose.
- [pytest-cobra](#)
  - PyTest plugin for testing smart contracts for Ethereum blockchain.

## Security Tools

- [MythX](#)
  - Security verification platform and tools ecosystem for Ethereum developers
- [Mythril](#)
  - Open-source EVM bytecode security analysis tool
- [Oyente](#)
  - Alternative static smart contract security analysis
- [Securify](#)
  - Security scanner for Ethereum smart contracts
- [SmartCheck](#)
  - Static smart contract security analyzer
- [Ethersplay](#)
  - EVM disassembler
- [Evmdis](#)
  - Alternative EVM disassembler
- [Hydra](#)
  - Framework for cryptoeconomic contract security, decentralised security bounties
- [Solgraph](#)
  - Visualise Solidity control flow for smart contract security analysis
- [Manticore](#)
  - Symbolic execution tool on Smart Contracts and Binaries
- [Slither](#)
  - A Solidity static analysis framework
- [Adelaide](#)
  - The SECBIT static analysis extension to Solidity compiler
- [solc-verify](#)
  - A modular verifier for Solidity smart contracts
- [Solidity security blog](#)
  - Comprehensive list of known attack vectors and common anti-patterns

- [Awesome Buggy ERC20 Tokens](#)
  - A Collection of Vulnerabilities in ERC20 Smart Contracts With Tokens Affected
- [Free Smart Contract Security Audit](#)
  - Free smart contract security audits from Callisto Network
- [Piet](#)
  - A visual Solidity architecture analyzer

## Monitoring

- [Alethio](#)
  - An advanced Ethereum analytics platform that provides live monitoring, insights and anomaly detection, token metrics, smart contract audits, graph visualization and blockchain search. Real-time market information and trading activities across Ethereum's decentralized exchanges can also be explored.
- [amberdata.io](#)
  - Provides live monitoring, insights and anomaly detection, token metrics, smart contract audits, graph visualization and blockchain search.
- [Neufund - Smart Contract Watch](#)
  - A tool to monitor a number of smart contracts and transactions
- [Scout](#)
  - A live data feed of the activities and event logs of your smart contracts on Ethereum
- [Tenderly](#)
  - A platform that gives users reliable smart contract monitoring and alerting in the form of a web dashboard without requiring users to host or maintain infrastructure
- [Chainlyt](#)
  - Explore smart contracts with decoded transaction data, see how the contract is used and search transactions with specific function calls
- [BlockScout](#)
  - A tool for inspecting and analyzing EVM based blockchains. The only full featured blockchain explorer for Ethereum networks.
- [Terminal](#)
  - A control panel for monitoring dapps. Terminal can be used to monitor your users, dapp, blockchain infrastructure, transactions and more.
- [Ethereum-watcher](#)
  - An extensible framework written in Golang for listening to on-chain events and doing something in response.
- [Alchemy Notify](#)
  - Notifications for mined and dropped transactions, gas price changes, and address activity for desired addresses.
- [Blocknatve Mempool Explorer](#)
- — Monitor any contract or wallet address and get streaming mempool events for every lifecycle stage — including drops, confirms, speedups, cancels, and more. Automatically decode confirmed internal transactions. And filter exactly how you want. Recieve events in our visual, no-code, interface or associate them with your API key to get events via a webhook. Mempool Explorer helps exchanges, protocols, wallets, and traders monitor and act on transactions in real-time.
- [Ethernal](#)
  - Ethereum block explorer for private chain. Browse transactions, decode function calls, event data or contract variables values on your locally running chain.

## Other Miscellaneous Tools

- [aragonPM](#)
  - a decentralized package manager powered by aragonOS and Ethereum. aragonPM enables decentralized governance over package upgrades, removing centralized points of failure.
- [Truffle boxes](#)

- 
  - Packaged components for building DApps fast.*[Cheshire](#)
- 
  - 
    - A local sandbox implementation of the CryptoKitties API and smart contracts, available as a Truffle Box
- [Solc](#)
- 
  - Solidity compiler
- [Sol-compiler](#)
- 
  - Project-level Solidity compiler
- [Solidity cli](#)
- 
  - Compile solidity-code faster, easier and more reliable
- [Solidity flattener](#)
- 
  - Combine solidity project to flat file utility. Useful for visualizing imported contracts or for verifying your contract on Etherscan
- [Sol-merger](#)
- 
  - Alternative, merges all imports into single file for solidity contracts
- [RLP](#)
- 
  - Recursive Length Prefix Encoding in JavaScript
- [eth-cli](#)
- 
  - A collection of CLI tools to help with ethereum learning and development
- [Ethereal](#)
- 
  - Ethereal is a command line tool for managing common tasks in Ethereum
- [Eth crypto](#)
- 
  - Cryptographic javascript-functions for Ethereum and tutorials to use them with web3js and solidity
- [Parity Signer](#)
- 
  - mobile app allows signing transactions
- [py-eth](#)
- 
  - Collection of Python tools for the Ethereum ecosystem
- [truffle-flattener](#)
- 
  - Concats solidity files developed under Truffle with all of their dependencies
- [Decode](#)
- 
  - npm package which parses tx's submitted to a local testrpc node to make them more readable and easier to understand
- [TypeChain](#)
- 
  - Typescript bindings for Ethereum smartcontracts
- [EthSum](#)
- 
  - A Simple Ethereum Address Checksum Tool
- [PHP based Blockchain indexer](#)
- 
  - allows indexing blocks or listening to Events in PHP
- [Purser](#)
- 
  - JavaScript universal wallet tool for Ethereum-based wallets. Supports software, hardware, and Metamask -- brings all wallets into a consistent and predictable interface for dApp development.
- [Node-Metamask](#)
- 
  - Connect to MetaMask from node.js
- [Solidity-docgen](#)
- 
  - Documentation generator for Solidity projects
- [Ethereum ETL](#)
- 
  - Export Ethereum blockchain data to CSV or JSON files

- [prettier-plugin-solidity](#)
  - Prettier plugin for formatting Solidity code
- [Unity3dSimpleSample](#)
  - Ethereum and Unity integration demo
- [Flappy](#)
  - Ethereum and Unity integration demo/sample
- [Wonka](#)
  - Nethereum business rules engine demo/sample
- [Resolver-Engine](#)
  - A set of tools to standarize Solidity import and artifact resolution in frameworks.
- [eth-reveal](#)
  - A node and browser tool to inspect transactions - decoding where possible the method, event logs and any revert reasons using ABIs found online.
- [Ethereum-tx-sender](#)
  - A useful library written in Golang to reliably send a transaction — abstracting away some of the tricky low level details such as gas optimization, nonce calculations, synchronization, and retries.
- [truffle-plugin-verify](#)
  - Seamlessly verify contract source code on Etherscan from the Truffle command line.
- [Blocknative Gas Platform](#)
- — Gas estimation for builders, by builders. Gas Platform harnesses Blocknative's real-time mempool data infrastructure to accurately and consistently estimate Ethereum transaction fees. This provides builders and traders with an up-to-the-moment gas fee API.
- [ETH Gas.watch](#)
  - A gas price watcher with email notifications on price change

## Smart Contract Standards & Libraries

### [ERCs](#)

- The Ethereum Request for Comment repository

- Tokens* [ERC-20](#)

  -
    - Original token contract for fungible assets

  - [ERC-721](#)

    -
      - Token standard for non-fungible assets

  - [ERC-777](#)

    -
      - An improved token standard for fungible assets

  - [ERC-918](#)

    -
      - Mineable Token Standard
- [ERC-165](#)
  - Creates a standard method to publish and detect what interfaces a smart contract implements.
- [ERC-725](#)
  - Proxy contract for key management and execution, to establish a Blockchain identity.
- [ERC-173](#)
  -

- A standard interface for ownership of contracts

## Popular Smart Contract Libraries

- [Zeppelin](#)
- 
  - Contains tested reusable smart contracts like SafeMath and OpenZeppelin SDK [library](#)
- for smart contract upgradeability
- [cryptofin-solidity](#)
- 
  - A collection of Solidity libraries for building secure and gas-efficient smart contracts on Ethereum.
- [Modular Libraries](#)
- 
  - A group of packages built for use on blockchains utilising the Ethereum Virtual Machine
- [DateTime Library](#)
- 
  - A gas-efficient Solidity date and time library
- [Aragon](#)
- 
  - DAO protocol. Contains [aragonOS smart contract framework](#)
- with focus on upgradeability and governance
- [ARC](#)
- 
  - an operating system for DAOs and the base layer of the DAO stack.
- [0x](#)
- 
  - DEX protocol
- [Token Libraries with Proofs](#)
- 
  - Contains correctness proofs of token contracts wrt. given specifications and high-level properties
- [Provable API](#)
- 
  - Provides contracts for using the Provable service, allowing for off-chain actions, data-fetching, and computation
- [ABDK Libraries for Solidity](#)
- 
  - Fixed-point (64.64 bit) and IEEE-754 compliant quad precision (128 bit) floating-point math libraries for Solidity

# Developer Guides for 2nd Layer Infrastructure

## Scalability

## Payment/State Channels

- [Ethereum Payment Channel](#)
- 
  - Ethereum Payment Channel in 50 lines of code
- [µRaiden Documentation](#)
- 
  - Guides and Samples for µRaiden Sender/Receiver Use Cases

## Plasma

- [Learn Plasma](#)
- 
  - Website as Node application that was started at the 2018 IC3-Ethereum Crypto Boot Camp at Cornell University, covering all Plasma variants (MVP/Cash/Debit)
- [Plasma MVP](#)
- 
  - OmiseGO's research implementation of Minimal Viable Plasma
- [Plasma MVP Golang](#)
- 
  - Golang implementation and extension of the Minimum Viable Plasma specification
- [Plasma Guard](#)
- 
  - Automatically watch and challenge or exit from Omisego Plasma Network when needed.
- [Plasma OmiseGo Watcher](#)
-

- Interact with Plasma OmiseGo network and notifies for any byzantine events.

## Side-Chains

- [POA Network](#)
- 
  - [POA Bridge](#)
- 
  - [POA Bridge UI](#)
- 
  - [POA Bridge Contracts](#)
- [Loom Network](#)
- [Matic Network](#)

## Privacy / Confidentiality

### ZK-SNARKs

- [ZoKrates](#)
- 
  - A toolbox for zkSNARKS on Ethereum
- [The AZTEC Protocol](#)
- 
  - Confidential transactions on the Ethereum network, implementation is live on the Ethereum main-net
- [Nightfall](#)
- 
  - Make any ERC-20 / ERC-721 token private - open source tools & microservices
- Proxy Re-encryption (PRE)
- **[NuCypher Network](#)
- 
  - A proxy re-encryption network to empower data privacy in decentralized systems
- **[pyUmbral](#)
- 
  - Threshold proxy re-encryption cryptographic library
- Fully Homomorphic Encryption (FHE)
- **[NuFHE](#)
- 
  - GPU accelerated FHE library

## Scalability + Privacy

## ZK-STARKs

- [StarkWare](#)
- and[StarkWare Resources](#)
- 
  - StarkEx scalability engine storing state transitions on-chain

## Prebuilt UI Components

- [aragonUI](#)
- 
  - A React library including Dapp components
- [components.bounties.network](#)
- 
  - A React library including Dapp components
- [ui.decentraland.org](#)
- 
  - A React library including Dapp components
- [dapparatus](#)
- 
  - Reusable React Dapp components
- [Metamask ui](#)
- 
  - Metamask React Components
- [DappHybrid](#)
- 
  - A cross-platform hybrid hosting mechanism for web based decentralised applications

- [Nethereum.UI.Desktop](#)
  - Cross-platform desktop wallet sample
- [eth-button](#)
  - Minimalist donation button
- [Rimble Design System](#)
  - Adaptable components and design standards for decentralized applications.
- [3Box Plugins](#)
  - Drop in react components for social functionality. Including comments, profiles and messaging.

info Inspired by:[https://github.com/ConsenSys/ethereum-developer-tools-list](https://github.com/ConsenSys/ethereum-developer-tools-list) [Edit this page](#) [Previous Launch Checklist](#)