---

title: Smart contract composability description: lang: en incomplete: true

---

A brief introduction {#a-brief-introduction}

Smart contracts are public on Ethereum and can be thought of as open APIs. You don't need to write your own smart contract to become a dapp developer, you just need to know how to interact with them. For example, you can use the existing smart contracts of Uniswap, a decentralized exchange, to handle all the token swap logic in your app – you don't need to start from scratch. Check out some of their v2 and v3 contracts.

What is composability? {#what-is-composability}

Composability is combining distinct components to create new systems or outputs. In software development, composability means developers can reuse existing software components to build new applications. A good way to understand composability is to think of composable elements as Lego blocks. Each Lego can be combined with another, allowing you to build complex structures by combining different Legos.

In Ethereum, every smart contract is a Lego of sorts—you can use smart contracts from other projects as building blocks for your project. This means you don't have to spend time reinventing the wheel or building from scratch.

How does composability work? {#how-does-composability-work}

Ethereum smart contracts are like public APIs, so anyone can interact with the contract or integrate them into dapps for added functionality. Smart contract composability generally works off three principles: modularity, autonomy, and discoverability:

1. Modularity: This is the ability of individual components to perform a specific task. In Ethereum, every smart contract has a specific use case (as shown in the Uniswap example).

2. Autonomy: Composable components must be able to operate independently. Each smart contract in Ethereum is self-executing and can function without relying on other parts of the system.

3. Discoverability: Developers cannot call external contracts or integrate software libraries into applications if the former are not publicly available. By design, smart contracts are open-source; anyone can call a smart contract or fork a codebase.

Benefits of composability {#benefits-of-composability}

Shorter development cycle {#shorter-development-cycle}

Composability reduces the work that developers have to do when creatingdapps. As Naval Ravikant puts it: "Open source means every problem has to be solved once."

If there is a smart contract that solves one problem, other developers can reuse it, so they don’t have to solve the same problem. This way, developers can take existing software libraries and add extra functionality to create new dapps.

Greater innovation {#greater-innovation}

Composability encourages innovation and experimentation because developers are free to reuse, modify, duplicate, or integrate open-source code to create desired results. As a result, development teams spend less time on basic functionality and can allocate more time experimenting with new features.

Better user experience {#better-user-experience}

Interoperability between components of the Ethereum ecosystem improves the user experience. Users can access greater functionality when dapps integrate external smart contracts than in a fragmented ecosystem where applications cannot communicate.

We'll use an example from arbitrage trading to illustrate the benefits of interoperability:

If a token is trading higher on `exchange A` than `exchange B`, you can take advantage of the price difference to make profit. However, you can only do that if you have enough capital to fund the transaction (i.e., buying the token from `exchange B` and selling it on `exchange A`).

In a scenario where you don't have enough funds to cover the trade, a flash loan might be idealFlash loans are highly technical, but the basic idea is that you can borrow assets (without collateral) and return same within *one* transaction.

Going back to our initial example, an arbitrage trader can take out a large flash loan, buy tokens from`exchange B`, sell them on `exchange A`, pay back the capital + interest, and keep the profit, within the same transaction. This complex logic requires combining calls to multiple contracts, which wouldn't be possible if smart contracts lacked interoperability.

Examples of composability in Ethereum {#composability-in-ethereum}

Token swaps {#token-swaps}

If you create a dapp that requires transactions to be paid in ETH, you can allow users to pay in other ERC-20 tokens by integrating token swap logic. The code will automatically convert the user's token to ETH before the contract executes the called function.

Governance {#governance}

Building bespoke governance systems for a DAO can be expensive and time-consuming. Instead, you could use an open-source governance toolkit, such as Aragon Client, to bootstrap your DAO to quickly create a governance framework.

Identity management {#identity-management}

Instead of building a custom authentication system or relying on centralized providers, you can integrate decentralized identity (DID) tools to manage authentication for users. An example is SpruceID, an open-source toolkit which offers a "Sign in with Ethereum" functionality that lets users authenticate identities with an Ethereum wallet.

Related tutorials {#related-tutorials}

- Contract Composability: The Building Blocks of Ethereum Smart Contract Development
- Kickstart your dapp frontend development with create-eth-app – *An overview of how to use create-eth-app to create apps with popular smart contracts out the box.*

Further reading {#further-reading}

Know of a community resource that helped you? Edit this page and add it!

- Composability is Innovation
- Why Composability Matters For Web3
- What is Composability?