What are Multi-Chain Contracts?

CosmWasm is designed and built from the ground up to be a multi-chain solution for smart contracts. As it comes from the Cosmos ecosystem, it is designed for networks of blockchains, rather than siloed chains. But what exactly do we mean by multi-chain?

Different Chain, Same Contract

Since we make a few requirements of the host application, it is easy for any Cosmos SDK app to embed thewasm module and customize the permissions or fees as they wish. All code is designed to be agnostic to the details of the underlying chain, so just by writing a CosmWasm contract, you will soon be able to run on different chains on the Cosmos ecosystem.

Regen Network plans to include CosmWasm support at launch. Several other chains are adding this support.

Inter Blockchain Contracts

If you have heard anything about Cosmos, it is most likely aboutInter-Blockchain Communication . The power ofTendermint BFT consensus and theirnovel bonded proof of stake algorithmare the foundation for a revolutionary protocol to allow trustless message-passing semantics between blockchains. No middleman, no timing issue, full security.

The potential means code on one chain can execute a transaction on another chain. But the code must be designed around a message-passing idiom. CosmWasm fully embraces theactor model and lends itself to IBC use. Messages are fire-and-forget, rather than awaiting a promise and worrying about race conditions and reentrancy attacks. As IBC stabilizes, we will be adding first-class support for IBC primitives into theCosmWasm libraries, as well as theCosmos SDK module that hosts it.

Easy to Integrate

Another design goal of CosmWasm was to be more of a library than a framework. This means it has a small surface area of required APIs and you can opt-in to most of the code. It is there to make life easy for you, but you can easily build it your own way as well.

This has two big benefits:

- It makes it easier to add support for multiple languages to write contracts in. So we can add support
- for say,AssemblyScript
- orGo
- , for those who
- prefer not to write in Rust.
- Since it makes limited demands on the host system, it can be embedded in other frameworks,
- not just the Cosmos SDK. The core runtime logiccosmwasm-vm
- is in Rust, andwasmvm
- provides a generic Go binding to it. As Go and
- Rust are two of the most popular languages to write blockchains, this opens the door for many integrations. Of course,
- unless your chain is running on top ofTendermint
- or potentially another BFT Instant Finality
- Consensus algorithm likeBabble
- , the contracts will not be able to interact via IBC.

Platform to Build On

CosmWasm doesn't want to lock you to one blockchain, or even one programming language. It is designed to be adaptable to many environments, andconnect blockchains. This makes it a solid platform to build on. Even if one chain doesn't pan out well, all your smart contracts and dApps can quickly be transferred to another chain. Or if your app grows quickly, you can launch your own chain to deploy the next version of the contracts, and transfer all existing tokens to your new chain via IBC. The possibilities are only limited by your imagination. Previous Next Steps Next Actor Model * Different Chain, Same Contract * Inter Blockchain Contracts * Easy to Integrate * Platform to Build On