Promises

Transactions can be sent asynchronously from a contract through aPromise . Like Promises in many programming languages, these will cause code to be executed in the future. In the case of NEAR, this "in the future" means a transaction to be executedin the next block (or thereabouts), rather than in the same block as the original function call.

You can implement any cross-contract workflow using Promises; they inhabit a middle-ground between the high-level and low-level approaches discussed inthe last section . See the full Promise docs, linked above, for details.

However, there are a few situations where Promises are uniquely capable, since these situations don't involve making function calls:

- Sending NEAR
- Creating accounts
- Deploying contracts

Why wait? Why not do these things synchronously, in the same block when the function is called? Why does NEAR require aPromise for sending tokens, or creating an account, or deploying a contract?

They need to be scheduled in separate blocks since sender and receiver accounts can be on different shards, and cross-shard communication happens across blocks by passing receipts (you can think of receipts in NEAR as "internal transactions"). You can see these receipts being passed from block to blockin NEAR Explorer . Edit this page Last updatedonDec 9, 2023 bygagdiez Was this page helpful? Yes No

Previous Callbacks Next Sending NEAR