Web3 Solana Blockchain - 05

So far, we have learned with solana/web3.js. We have been able to connect with the Solana blockchain, create a keypair and perform a simple transaction. Let us take the next steps and look into interacting with Programs on Solana.

Program

In simple terms, a **program** consists of a few lines of code that can interpret instructions. An instruction specifies which program it is calling, which accounts it wants to read or modify, and additional data that serves as an auxiliary input to the program.

To understand programs, you need to understand a little bit about how programs work and how they are integrated into Dapps.

The workflow to interact with Solana programs is as follows:

1. An application interacts with a Solana cluster by sending it transactions with one or more instructions.
2. The Solana runtime passes those instructions to programs deployed by app developers beforehand. An instruction might, for example, tell a program to transfer lamports from one account to another or create an interactive contract that governs how lamports are transferred.
3. In our example, we are going to send an instruction that ***says hello*** to the deployed Solana program.
4. Instructions are executed sequentially and atomically for each transaction. If any instruction is invalid, all account changes in the transaction are discarded.

Sanity tests

Sanity tests, also known as smoke tests, are a type of software testing that focuses on quickly and superficially checking the basic functionality of an application or system. The primary goal of sanity testing is to determine whether the application or system is fundamentally working as expected or if there are any major issues that would prevent further testing or usage.

Sanity tests are typically performed after a software build or release to ensure that the critical functionalities are working correctly. These tests are usually executed without detailed testing scenarios or exhaustive test cases. Instead, they aim to cover essential features and use cases, providing a basic level of confidence in the system's stability.

The purpose of sanity testing is to identify major issues or regressions that could disrupt normal operation, rendering the application or system unusable or significantly impacting its core functionality. These tests can

help catch severe issues early in the testing process, allowing the development team to quickly address them before proceeding with more comprehensive testing.

Sanity tests often include verifying critical functionalities, such as logging in, accessing key system components, performing basic operations, and checking for obvious errors or abnormalities. They serve as a quick check to determine if the system is in a reasonable state for further testing and validation.

It's important to note that sanity tests are not meant to be exhaustive or replace thorough testing methodologies like functional testing, regression testing, or performance testing. They provide a high-level assessment of the system's stability, ensuring that critical features are intact before moving on to more comprehensive testing activities.