ANASTASIA LABS

Proof of Achievement - Milestone 3

Documentation Refinement and Technical Specifications

Project Number 1100024
Project Manager Jonathan Rodriguez

Contents

Project Name: Lucid Evolution: Redefining Off-Chain Transactions in Cardano
URL: Catalyst Proposal

Introduction

Since our Milestone 2 submission, we've been working intensively on expanding Lucid Evolution's functionality, with our commit count now approaching 900. Our focus has been on:

- Transitioning to PlutusV3
- Being Chang Hardfork ready
- Adapting to newly altered cost structures
- Updating our Tx-Builder library to align with the current state of Cardano

The post-Chang Hardfork period has seen popular dApps undergoing a transition phase. This has given us the opportunity to collaborate closely with many developers, actively assisting with errors and queries, fostering discussions around pain points, and adapting our library based on their needs. We've ensured our documentation is comprehensive, covering all the backend work we've been busy with. Lucid Evolution has matured significantly and is now ready for use by any developer in our ecosystem. We've maintained a syntax familiar to users of the legacy Lucid library, aiming to create a product that's:

- Easy to use out-of-the-box
- Familiar to many in the community
- Designed to simplify developers' lives with their off-chain operations, rather than introducing new complexities

Our goal has been to provide a library that eases the development process, particularly for those already acquainted with Lucid's approach to off-chain operations in the Cardano ecosystem

Documentation Website

To make it easy for developers to follow our documentation, we've created a centralized knowledge base. This comprehensive resource serves as a go-to place for developers to lookup examples, check use-cases, review syntax they might have forgotten, and gain a general understanding of Lucid Evolution and how it can be used in various scenarios.

We've put considerable effort into making the format and structure as user-friendly as possible. Our goal is to ensure that even someone with no prior knowledge of Lucid Evolution can start using it quickly and efficiently. The navigation is intuitive, allowing users to find the information they need without unnecessary complexity.

We are excited to introduce our new website to the Catalyst reviewers:

- https://anastasia-labs.github.io/lucid-evolution

This website represents the long-term refinement and community collaboration. We've been working on improving it for an extended period, gathering and incorporating feedback from developers in our community. Their input has been invaluable in shaping the content and improving its overall usability.

In the spirit of open collaboration, we've encouraged a system that allows fellow developers to suggest changes to the knowledge base we are creating. We provide templates for different topics, encouraging contributions and ensuring that the documentation evolves with the needs of our user base.

At this stage, we believe the documentation has reached a polished state and covers almost the entire functionality of the library. However, our work is far from over. We are actively expanding the site with new use case scenarios and providing more examples to interested readers. This ongoing effort is so that the documentation remains a living, growing resource that adapts to the evolving needs of our developer community.

Our commitment to comprehensive, user-friendly documentation reflects our broader goal: to make Lucid Evolution not just a powerful tool, but an accessible one. We believe that by providing clear, thorough, and practical documentation, we can empower more developers in our ecosystem.

Documentation Structure for Lucid Evolution

The documentation for Lucid Evolution is organized into several key sections to facilitate ease of use and understanding for developers. A rough structure of the documentation looks like this excluding our provided `Examples`, `Conway Era` functions usage, and further deep dives like tx-chaining:

Introduction

Overview of Lucid Evolution and its purpose. Key features and benefits for developers.

Installation

Step-by-step guide on how to install Lucid Evolution using various package managers (npm, yarn, pnpm, Bun). Compatibility information and prerequisites.

Core Concepts

- Instantiate Evolution: Instructions on how to set up the Lucid instance.
- Choose Wallet: Guidance on selecting and managing wallets.
- Your First Transaction: A walkthrough for building and submitting a basic transaction.

Deep Dives

In-depth guides on specific functionalities such as minting assets, smart contract interactions, and using the emulator.

- **Make Payments**: Overview of various methods for making payments. Examples for simple ADA payments and multiple recipients. Code snippets demonstrating the payment process.
- **Mint Assets**: Instructions on creating a minting policy for assets. Steps to derive the policy ID from the minting policy script. Code examples for minting tokens and attaching the minting policy.
- **Smart Contract Interactions**: Explanation of how to interact with validators instead of traditional smart contracts. Steps to instantiate validators and work with datums and redeemers. Code snippets illustrating the interaction process.
- **Emulator**: Introduction to the emulator for testing and validating transactions in a controlled environment.Steps to initialize the emulator and work with predefined addresses and asset distributions. Code examples for simulating transactions and distributing staking rewards.
- **Register Stake**: Instructions on how to register a stake address and manage staking operations.
- **Register/Retire Stakepool**: Guidelines for registering and retiring a stake pool.

Syntax Change Documentation - From Legacy Lucid to Lucid Evolution

Overview

We've made some exciting changes in Lucid Evolution, and we want to make sure everyone is up to speed. This section highlights the key differences between the legacy Lucid library and the new Lucid Evolution. We've aimed to keep things familiar while introducing some powerful new features.

You can find all these highlights (and more) on our official documentation page: https://anastasia-labs.github.io/lucid-evolution

Core Concepts

Instantiating Lucid

Changes: We've simplified things a bit here. Instead of using `Lucid.new()`, you now just call `Lucid()`. It's a factory function now, not a class.

Check it out: https://anastasia-labs.github.io/lucid-evolution/documentation/core-concepts/instantiate-evolution

Creating or Choosing a Wallet

Changes: We've made the syntax a bit more intuitive. Instead of `selectWalletFromPrivateKey`, you now use `selectWallet.fromPrivateKey`.

Learn More https://anastasia-labs.github.io/lucid-evolution/documentation/core-concepts/choose-wallet

Your First Transaction

The Difference: We've renamed `payToAddress` to `pay.ToAddress`. It's a small change, but we think it makes things clearer.

Deep Dives

Making Payments

Changes:

- `payToAddress` is now `pay.ToAddress`
- `payToAddressWithData` has become `pay.ToAddressWithData`

The `pay.ToAddressWithData` method now accepts these parameters:

1. Address
2. OutputDatum (can be "hash", "inline", or "asHash")
3. Assets (optional)
4. Script reference (optional)

Check it out: https://anastasia-labs.github.io/lucid-evolution/documentation/deep-dives/make-payments

Minting Assets

Changes:

- `attach.MintingPolicy` is now `attachMintingPolicy`
- We've introduced a new function for complex script parameters: `applyDoubleCborEncoding`

When you're working with complex script parameters (like nested structures or custom data types), you'll need to use `applyDoubleCborEncoding` before applying the parameters to the script. Here's how it looks:

```
1  const scriptWithParams = applyParamsToScript(
2    applyDoubleCborEncoding(yourScript),
3    [param1, param2, ...]
4  );
```

Important Note: Remember, minting tokens creates them, but it doesn't automatically assign them to an address. After minting, the tokens are owned by the transaction itself. That's why you'll often see `pay.ToAddress()` used right after minting - it's sending those newly minted tokens to a specific address.

Learn More: https://anastasia-labs.github.io/lucid-evolution/documentation/deep-dives/mint-assets

—

We hope this guide helps you find the most important syntax changes from the legacy Lucid to Lucid Evolution in one look. As always, we're here to help if you have any questions. Happy coding!