ANASTASIA LABS

Proof of Achievement - Milestone 1

PR Implementations in Lucid - Evolution

Project Number	1100024
Project Manager	Jonathan Rodriguez

ANASTASIA LABS

Contents

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

Evolution Implementations of the PRs from legacy Lucid

Evaluation Methodology

While transitioning the legacy Lucid library to Lucid-Evolution, we didn't just implement each and every standing pull request (PR) in the legacy repository; we scrutinized each one to see if it fit the Evolution setup.

Lucid-Evolution library was created from scratch and has had an extensive development journey already. We would like to point out the **"Closed PRs Overview"** section of this report where we showcase the progress we have made outside the scope of standing PRs from the legacy library as well.

In open-source software, a PR is essentially a feature request from the community, not a must-have, and should be evaluated for its relevance and necessity at different stages of development.

Our approach was straightforward: assess each PR on its own merits, asking whether it serves the library's current goals and technical needs.

We can break down our approach into a few criteria:

- **Relevance:** Does this feature make sense for Lucid-Evolution right now?
- **Technical Fit:** Can we integrate it smoothly, and does it help us in the long run?
- **Community Feedback:** How much do users and developers need the feature?

This method helped us make sure we're building a library that's not just powerful but also what our community needs and wants.

Summary of Decisions (Table)

PR Nr.	Title	Status
234	Stop Leaking Memory	**Implemented**
233	Fix memory leaks from Tx	**Implemented**
231	Feat: chaining of txs	**Implemented**
220	Feat (Next.JS): Support server-side Next.JS usage	**Implemented**
219	Add Koios Provider to Lucid	**Implemented**
218	Prototype Hydra provider	Not Implemented
217	Wallet from seed should query both base and enterprise addresses on signing	Not Implemented
197	Fix typebox portable type annotation	**Implemented**
111	Add graphql provider	Not Implemented

Detailed Implementation of Each PR

In the following section, you will find the implemented solutions to the standing PRs from the legacy repository that were evaluated as relevant for the design of Lucid-Evolution.

Under the "Implemented" header you can find the the PRs that are included in Lucid-Evolution with the solution description provided together with information to display which PR they correspond to in the legacy repository.

Implemented

Fix typebox portable type annotation

PR Nr.	Date	Title	Submitter	Link
197	Jun 7, 2023	Fix typebox portable type annotation	solidsnakedev Link	GitHub PR

Description / Purpose

Building on the foundational work done in the previous Lucid library, the Lucid-Evolution update has significantly advanced our approach to handling type annotations, ensuring greater portability and compatibility across different environments.

Background

The original PR aimed to fix a critical issue in the TypeScript configuration that affected the naming of inferred types like 'CredentialSchema'. This problem was particularly challenging as it hindered the deployment flexibility of the project due to dependencies on specific modules in node_modules.

Evolutionary Improvements

In Lucid-Evolution, we have expanded upon the initial intentions of the PR by implementing a more robust and flexible system for defining and handling data types. This is achieved through the integration of TypeBox, which allows us to define types with precise constraints and conditions.

Further Details to our Implementation

GitHub Blob

Technical Implementation

```
import * as TypeBox from "@sinclair/typebox";
import { Data } from "@lucid-evolution/core-types";

export const Data = {
  Enum: function <T extends TypeBox.TSchema>(items: T[]) {
    const union = TypeBox.Type.Union(items);
    replaceProperties(union, {
      anyOf: items.map((item, index) =>
        item.anyOf[0].fields.length === 0
          ? {
              ...item.anyOf[0],
              index,
            }
          : {
              dataType: "constructor",
              title: item.anyOf[0].fields[0].title,
              index,
              fields:
                item.anyOf[0].fields[0].items ||
                item.anyOf[0].fields[0].anyOf[0].fields,
            },
      ),
    });
    return union;
  },
};
```

Impact

The updated implementation not only addresses the portability issues but also enhances the overall architecture by allowing more explicit and clear type declarations. This ensures that the system remains robust in diverse deployment scenarios, thus eliminating previous barriers that might affect the deployment process.

Add Koios Provider to Lucid

PR Nr.	Date	Title	Submitter	Link
219	Sep 3, 2023	Add Koios Provider to Lucid	edridudi	GitHub PR

Background

Building upon the foundational work done in PR 219 of the legacy Lucid library, the Lucid-Evolution library further enhances and integrates the capabilities of the Koios API. The original PR introduced the Koios Provider, enabling data retrieval capabilities for the Lucid library, which we have now expanded and refined.

Evolutionary Improvements

In Lucid-Evolution, the Koios provider has been integrated and updated to better match the needs of the Cardano blockchain environments, including Mainnet, Preview, and Preprod. This ensures that our library can support a wider range of applications with increased reliability and efficiency.

Impact

The integration and enhancement of the Koios provider in Lucid-Evolution not only preserve the functionalities introduced in the legacy Lucid library but also enhance them to offer more robust and versatile tools for developers. This implementation supports the library's aim to provide streamlined, effective interactions with the Cardano blockchain, improving both developer experience and application performance.

Technical Implementation

Here is a look at how the Koios provider has been adapted and implemented in the Lucid-Evolution library:

```
import { Koios } from "@lucid-evolution/provider/koios";

// Enhanced initialization for varied network support
const koiosProvider = new Koios("https://api.koios.rest/api/v1");

// Advanced method to fetch protocol parameters with error
handling
async function fetchProtocolParameters() {
  try {
    return await koiosProvider.getProtocolParameters();
  } catch (error) {
    console.error("Error fetching protocol parameters: ", error);
  }
}

// Expanded UTXO retrieval capabilities
wallet.getUtxos = async (address) => {
  try {
    return await koiosProvider.getUtxos(address);
  } catch (error) {
    console.error("Error retrieving UTXOs: ", error);
  }
};
```

Further Details to our Implementation

GitHub PR [Evolution]

Feat (Next.JS): Support serverside Next.JS usage

PR Nr.	Date	Title	Submitter	Link
220	Sep 6, 2023	Feat (Next.JS): Support serverside Next.JS usage	thaddeusdiamond	GitHub PR

Background

Building on PR 220 from the legacy Lucid library, Lucid-Evolution has further refined the server-side usage of Next.js to address compatibility issues, particularly those introduced in Next.js version 12.0.1 concerning relative imports. This enhancement ensures that Lucid-Evolution can operate seamlessly in Next.js environments, adhering to the latest standards and practices.

Evolutionary Improvements

In Lucid-Evolution, the integration of Next.js has been meticulously enhanced to not only detect the Next.js environment but also to handle import paths dynamically, ensuring compatibility and efficiency in server-rendered applications. This addresses the key issues identified in related PR 174, which highlighted bugs due to Next.js's handling of imports.

Technical Implementation

You can see below is a snippet showcasing the integration approach, focusing on handling Next.js environments within configuration files:

```
const { resolve } = require("node:path");
const project = resolve(process.cwd(), "tsconfig.json");

/** @type {import("eslint").Linter.Config} */
module.exports = {
  extends: [
    "@lucid-evolution/eslint-config/next.js"
  ],
  parser: "@typescript-eslint/parser",
  parserOptions: {
    project: project,
    // Additional ESLint rules to handle Next.js specific
scenarios
  },
};
```

Further Details to our Implementation

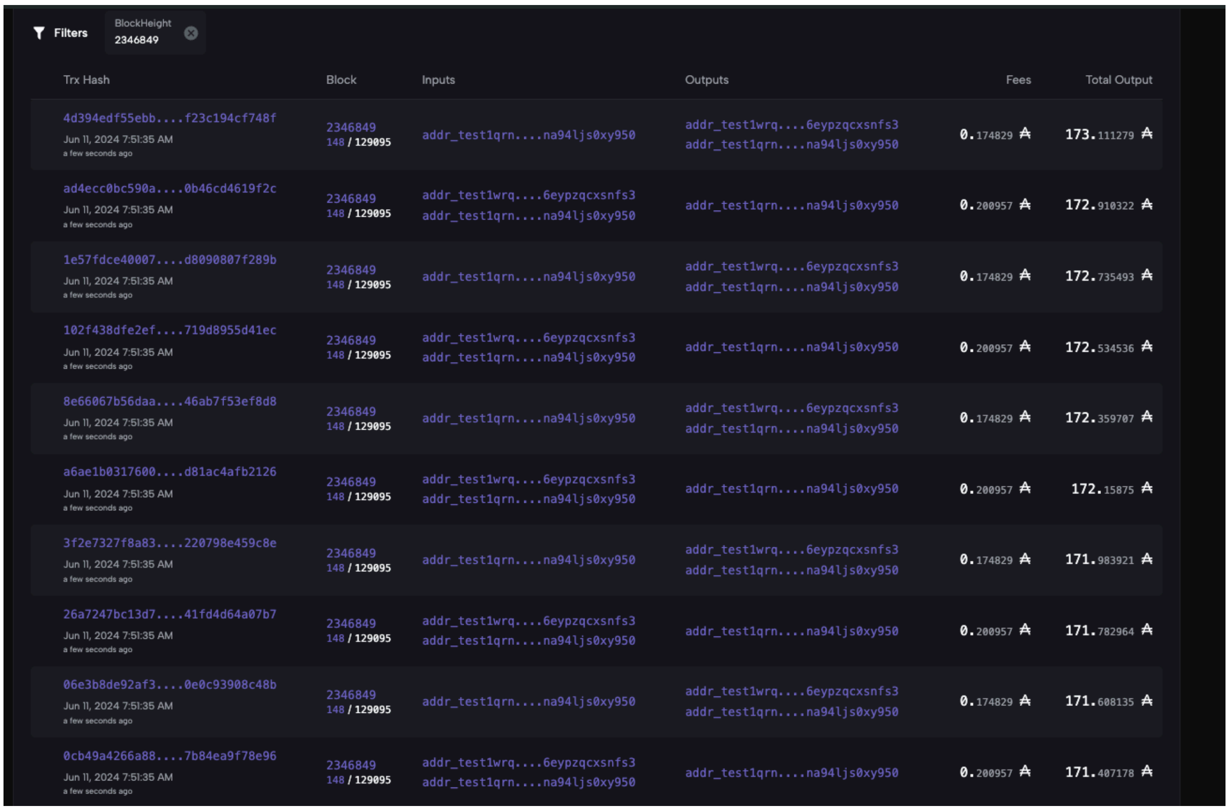
GitHub Search [Evolution]

Feat: chaining of txs

PR Nr.	Date	Title	Submitter	Link
231	Oct 30, 2023	Feat: chaining of txs	will-break-it	GitHub PR

This feature that has been successfully implemented in the Evolution library is a good example serving as a demonstration of how the purpose of Lucid - Evolution is not only to maintain but also to extend and supersede the legacy Lucid library in terms of features requested by the community

In our Discord Channel exemplary usage of this feature has already been showcased and it can also be found in the latest `lucid-evolution` release `0.2.40`. The following image displayed is an example of how 10 transactions were fit in one block using Blockfrost via our txchaining method:



Trx Hash	Block	Inputs	Outputs	Fees	Total Output
4d394edf55ebb....f23c194cf748f Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1qrn....na94ljs0xy950	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	0.174829 ₳	173.111279 ₳
ad4ecc0bc590a....0b46cd4619f2c Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	addr_test1qrn....na94ljs0xy950	0.200957 ₳	172.910322 ₳
1e57fdce40007....d8090807f289b Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1qrn....na94ljs0xy950	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	0.174829 ₳	172.735493 ₳
102f438dfe2ef....719d8955d41ec Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	addr_test1qrn....na94ljs0xy950	0.200957 ₳	172.534536 ₳
8e66067b56daa....46ab7f53ef8d8 Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1qrn....na94ljs0xy950	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	0.174829 ₳	172.359707 ₳
a6ae1b0317600....d81ac4afb2126 Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	addr_test1qrn....na94ljs0xy950	0.200957 ₳	172.15875 ₳
3f2e7327f8a83....220798e459c8e Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1qrn....na94ljs0xy950	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	0.174829 ₳	171.983921 ₳
26a7247bc13d7....41fd4d64a07b7 Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	addr_test1qrn....na94ljs0xy950	0.200957 ₳	171.782964 ₳
06e3b8de92af3....0e0c93908c48b Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1qrn....na94ljs0xy950	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	0.174829 ₳	171.608135 ₳
0cb49a4266a88....7b84ea9f78e96 Jun 11, 2024 7:51:35 AM a few seconds ago	2346849 148 / 129095	addr_test1wrq....6eypzqcxsnfs3 addr_test1qrn....na94ljs0xy950	addr_test1qrn....na94ljs0xy950	0.200957 ₳	171.407178 ₳

Figure 1: tx chaining via Blockfrost (Testnet)

Background

Following the intent of PR 231 from the legacy Lucid library, the Lucid-Evolution framework has significantly enhanced the capabilities for transaction chaining. This feature allows for the efficient chaining of multiple transactions within a single block, addressing scalability and performance enhancements critical for modern blockchain applications.

Evolutionary Improvements

The enhancements in Lucid-Evolution include sophisticated management of wallet UTXOs and outputs derived from transactions. This ensures that transactions can be chained reliably without waiting for previous transactions to be recorded on the blockchain, which is pivotal for applications requiring rapid sequential transactions.

Technical Implementation

The implementation of transaction chaining involved several key developments:

- The `TxBuilder` class has been expanded to support chaining by maintaining a state of wallet inputs and outputs across chained transactions. This allows for seamless transition from one transaction to another.
- The `overrideUTxOs` method has been introduced to dynamically adjust the wallet's UTXOs, enabling the wallet to use unconfirmed outputs as inputs for subsequent transactions.
- The `chain`, `chainProgram`, and `chainSafe` methods in `TxBuilder` facilitate the construction and submission of chained transactions by handling the complexities of UTXO management and transaction construction within the framework.

Demonstration via Syntax

A code snippet demonstrating how transaction chaining is implemented:

```
const lucid = await Lucid.init(config);
const [newWalletInputs, derivedOutputs, tx] = await lucid.newTx()
  .payToAddressWithData(address, {kind: "inline", value: datum},
{lovelace: 10000000n})
  .chain();

const signedTx = await tx.sign().complete();
const txHash = await signedTx.submit();
lucid.overrideUTxOs(newWalletInputs);
```

Lucid Evolution

Release 0.2.40

Further Details to our Implementation

GitHub PR [Evolution]

Fix memory leaks from Tx

PR Nr.	Date	Title	Submitter	Link
233	Nov 3, 2023	Fix memory leaks from Tx	joacohoyos	GitHub PR

Stop Leaking Memory

PR Nr.	Date	Title	Submitter	Link
234	Nov 4, 2023	Stop Leaking Memory	yHSJ	GitHub PR

Background

Building on the efforts made in PRs 233 and 234 in the legacy Lucid library, Lucid-Evolution has adopted significant memory management improvements, primarily facilitated through updates in the Cardano Multiplatform Library (CML) by DcSpark.

These updates were essential to address persistent memory leaks within transaction processing and general library operations, particularly when interfaced with JavaScript in WASM builds.

Evolutionary Improvements

Lucid-Evolution leverages the enablement of weak references in CML's WASM builds to address the memory leak issues described in PRs 233 and 234:

- The introduction of weak references allows for automatic memory management by the garbage collector, reducing the need for explicit free() calls and improving overall system stability and performance.
- Changes in CML, including the optimization of memory handling and enabling weak references, have directly impacted Lucid-Evolution's memory management capabilities, ensuring that memory leaks are minimized even under heavy allocation loads.

Technical Implementation

The technical changes from the legacy PRs focused on overhauling memory handling within the Tx class and refining the usage of data structures interfacing with WASM.

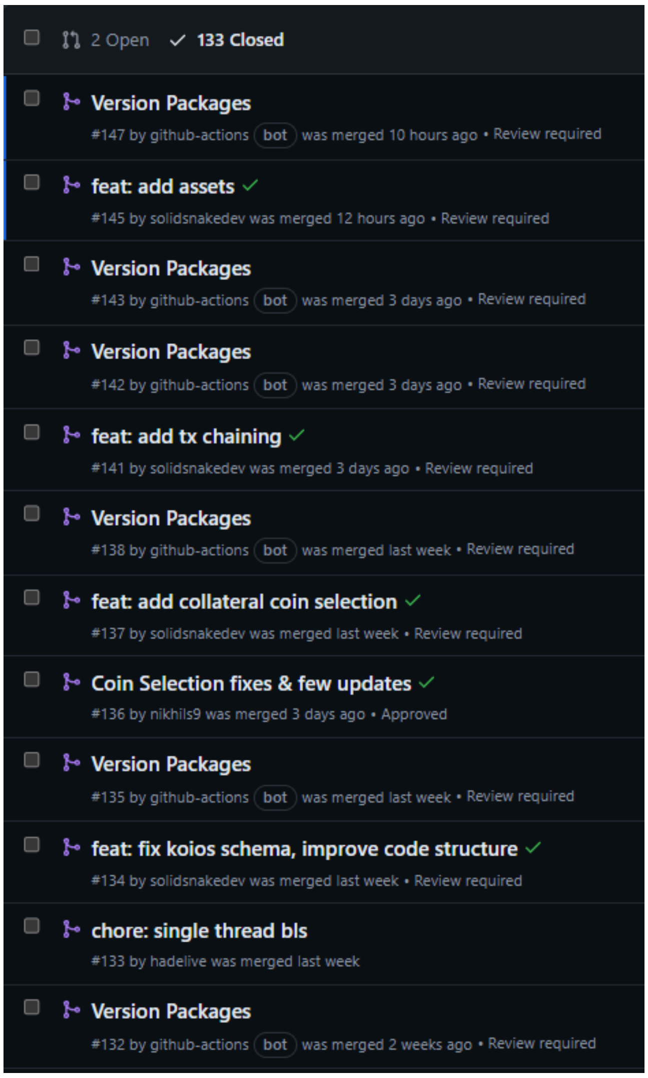
Further Details to CML

Weakrefs in WASM builds

ANASTASIA LABS

Closed PRs Overview

We have closed numerous PRs (**over 130 already**) in the Lucid Evolution repository, showcasing extensive development work. The exhaustive list of PRs can be found in the Github repository and also can be accessed here



2 Open ✓ 133 Closed

Version Packages
#147 by github-actions bot was merged 10 hours ago • Review required

feat: add assets ✓
#145 by solidsnakedev was merged 12 hours ago • Review required

Version Packages
#143 by github-actions bot was merged 3 days ago • Review required

Version Packages
#142 by github-actions bot was merged 3 days ago • Review required

feat: add tx chaining ✓
#141 by solidsnakedev was merged 3 days ago • Review required

Version Packages
#138 by github-actions bot was merged last week • Review required

feat: add collateral coin selection ✓
#137 by solidsnakedev was merged last week • Review required

Coin Selection fixes & few updates ✓
#136 by nikhils9 was merged 3 days ago • Approved

Version Packages
#135 by github-actions bot was merged last week • Review required

feat: fix koios schema, improve code structure ✓
#134 by solidsnakedev was merged last week • Review required

chore: single thread bls
#133 by hadelive was merged last week

Version Packages
#132 by github-actions bot was merged 2 weeks ago • Review required

Not Implemented

Add graphql provider

PR Nr. (Legacy)	Date	Title	Submitter	Link
111	Oct 29, 2022	Add graphql provider	zachykling	GitHub PR

Wallet from seed should query both base and enterprise addresses on signing

PR Nr.	Date	Title	Submitter	Link
217	Aug 28, 2023	Wallet from seed should query both base and enterprise addresses on signing	infrmtcs	GitHub PR

Prototype Hydra Provider

PR Nr.	Date	Title	Submitter	Link
218	Aug 29, 2023	Prototype Hydra provider	Piefayth	GitHub PR

- we assessed this PR and we noticed that this graphgl is outside of the scope of a tx builder . External cardano technology too specific for our scope.
- seed should query
- Prototype Hydra Provider is still ongoing therefore its not materialized it wil be included at the stage of development for milestone 4.