TxPipe Docs

# Tx3 Language Guide

This guide provides a comprehensive overview of the Tx3 language, its syntax, and features.

## Protocol

A Tx3 file describes a UTxO protocol. By protocol we mean the set of conventions required to construct the transactions that allows a user to interact with the blockchain to fulfill a goal.

Another common term for protocol is a dApp, but we prefer to call them protocols to decouple it from relationship to UI artifacts. Tx3 does NOT provide a way to build frontends, but it does simplify a lot of the boilerplate that these UIs need to integrate blockchain operations.

## Parties

The first thing you need to specify in your protocol is the parties involved in the interactions.

A party is an abstraction for someone or something interacting with your protocol. In technical terms, a party is just a placeholder for an on-chain address.

**Party Syntax**

Details on how to specify and use Parties

# Transactions

The main components of a protocol are the transaction template definitions.

A transaction describes an action that the protocol can execute. A template for transaction resembles a *function* in the sense that it takes some arguments and returns a concrete transaction that can be submitted to the blockchain.

The body of a transaction template in Tx3 needs to fully describe the structure of an UTxO transaction. This includes things like inputs, outputs and how they connect to each other.

**Transaction Syntax**

Details on how to specify and use Transaction Templates

# Policies

Policies represent the specific logic involved with a protocol. These are usually scripts on-chain that validate actions.

Policies are used by the protocol to describe rules that govern how certain operations work. Operations like spending UTxOs, minting assets, etc.

Policies can also act as a special kind of party of the protocol.

**Policy Syntax**

Details on how to specify and use Policies

# Types

Tx3 has a set of primitive types but it also support the definition of custom types when required.

Types are used by the compiler to know how to serialize data structures when generating the concrete transactions. Types are also used to catch early errors during the static analysis of the protocol.

**Types** →

Details on how to specify and use Types

# Data Expressions

Tx3 supports data expression to describe and operate with data structures.

Data expressions are used whenever a protocol needs to express a value in term of parameters, the result of a computation or by constructing specific types of data.

The contents of datums and redeemers are expressed as data expressions.

**Data Expressions** →

Details on how to specify and use Data Expressions

# Asset Expressions

Tx3 defines specific rules for expressing assets in UTxO's value.

Operating with quantities of different asset classes using normal data expressions is too verbose. Tx3 introduces specific asset expressions to simplify this task. Asset expressions provide basic math operators that can be applied to multi-class assets in a very natural way.

**Asset Expressions** →

Details on how to specify and use Asset Expressions

# Chain-specific Features

Tx3 includes ad-hoc language artifacts to deal with constructs that are specific to a particular blockchain.

**Cardano Features** →

Details on how to specify and use Cardano-specific features