ZEROCODEX

Knowledge that can execute

An open standard for portable data meaning, computation and assumptions

ZeroCodex turns knowledge normally buried in spreadsheets, scripts, APIs, dashboards and individual expertise into reusable executable Recipes.

The problem is not just data

Organizations and researchers already have enormous amounts of data. What is repeatedly lost is the knowledge required to use it: which source is authoritative, how fields correspond, how a result is calculated, which choices were made, and what should happen when those choices change. ZeroCodex places that knowledge in a reusable layer above the underlying systems.

Recipes

A Recipe is an executable description of how to obtain or produce something useful. It can read existing sources, call other Recipes, transform data, return results, and expose the choices that matter. Recipes are graphs; 0CX is their compact canonical textual form.

A Recipe can be tiny:

```
42
"Vancouver"
recipe.0cx
```

Or it can describe a large graph joining many sources and other Recipes. Complexity is added only when the problem requires it.

Assumptions are first-class

ZeroCodex distinguishes ordinary constants from Assumptions: named values that a Recipe deliberately invites downstream users to inspect or change. An assumption might be a threshold, exchange rate, interpolation method, confidence rule, business definition or visualization parameter.

- Assumptions inherit. Downstream Recipes can use choices made upstream.
- Assumptions can be overridden. Users can ask “what if?” without rewriting the original Recipe.
- Dependencies remain visible. A changed Assumption can recompute the affected graph instead of rebuilding the analysis.

This makes experimentation, auditing and scenario analysis natural properties of the same executable object.

A common semantic layer

ZeroCodex does not require data to be moved into a universal database. A Codex maps local reality onto shared concepts. CityCodex does this for civic and public data. CorpCodex does the same for organizational and business knowledge. Domain Recipes can then operate on standardized meaning rather than each source system's storage details.

From calculation to composition

The same Recipe model scales from a number to an interactive system

The important unit in ZeroCodex is not an application or database. It is a reusable Recipe that can be called, composed, questioned, published and executed in different environments.

Portable analysis

Once a domain has a standardized Codex, generic Recipes become portable. A business-analysis Recipe for customer concentration or inventory turns can be applied to different companies; a civic Recipe can be applied to different municipalities. The local Codex handles translation while the analytical Recipe carries reusable expertise.

Consultants can work against published organizational meaning instead of repeatedly reconstructing definitions from exports and interviews - and can leave their own useful analysis behind as Recipes.

Modifiers and animation

An Assumption does not have to be changed only by a person. A Modifier can drive it from another value: population can scale a symbol, confidence can control opacity, geography can change colour, or another dataset can change a threshold. Animation is simply a useful Modifier whose driver includes animation time.

ZeroCodex can distinguish data time from presentation time. Historical state can remain fixed while its presentation animates, or a time-aware Recipe can advance through years, seconds or finer precision.

Recurring and event-driven Recipes

In organizational use, Recipes can be activated by schedules, events or conditions while the trigger remains separate from the business logic.

- Every morning: calculate yesterday's sales, make a chart, and publish it.
- Inventory below threshold: evaluate stock and purchasing Recipes, then take the defined action.
- Invoice state changes: run the same deterministic follow-up logic used everywhere else.

This supports deterministic agents: persistent automation whose decisions and actions are expressed through inspectable, versioned Recipes rather than hidden in one-off workflow scripts.

AI can help without becoming the Runtime

AI can be useful in the Kitchen: identifying columns, interpreting unfamiliar sources, proposing mappings, following references, or drafting a Recipe. But the accepted Recipe can remain deterministic, executable without an LLM, and inspectable, testable, versioned and reproducible.

The Kitchen and the Runtime

The Kitchen is the visual authoring environment. Researchers, analysts and developers can work primarily with graphs rather than hand-written 0CX. The Runtime executes the canonical Recipe. There is no need for a second private representation to carry its meaning.

A Recipe can also be a publication

Presentation is optional - but it can travel with the computation

A Recipe can return data or computation alone. When presentation is included, the same Recipe can also become a rich interactive publication without becoming a separate web application.

Presentation as a top-level capability

A Recipe may optionally describe how its result should be presented: explanatory text, maps, charts, highlighted Assumptions, controls and viewer behavior. For most authors this is semantic markup, not page layout. The Kitchen can provide a simple rich-text editor while storing a portable Markdown-based presentation.

Future HTML/CSS can provide an expert escape hatch without changing the Recipe's computational meaning. The standard viewer remains responsible for connecting presentation controls to the live graph.

Interactive research becomes cheap to publish

A researcher can publish an analysis together with chosen source datasets and selected Assumptions. Readers can see the result immediately, change the assumptions the author exposes, and recompute in the browser. Advanced users can open the same Recipe in the Kitchen and inspect the full graph.

- The publication can be hosted as ordinary static files.
- Computation and visualization can run client-side where practical.
- The publisher does not need to build or maintain a bespoke interactive web application.
- The Recipe remains reusable outside the publication that introduced it.

0CX: language, artifact and address

0CX is the language and .0cx is the Recipe file extension. Public Recipes can have a deliberately simple canonical web identity:

```
recipe.0cx   + .cx   →   recipe.0cx.cx
```

The web address can open a rich viewer for the same Recipe: summary, demonstration data, interactive Assumptions, provenance, related Recipes and an Open in Kitchen path. It is a human-facing view of the executable artifact, not a separate publication.

Open by design

ZeroCodex.org is the home of the open standard, documentation and community. 0cx.cx is the natural public namespace for 0CX Recipes and their interactive views. The Runtime, file format and resolution rules are intended to remain open so Recipes are not dependent on one commercial service.

Enterprise services can exist around that open foundation - private infrastructure, identity, permissions, connectors, governance, auditing, scheduled execution, compliance and support - without withholding the core language, viewer or Recipe model.

The larger idea

ZeroCodex aims to make meaning as reusable as data. A definition created once can be called elsewhere. An assumption can remain visible instead of disappearing into a script. An analysis can become a Recipe instead of a dead report. A visualization can carry the computation that produced it. And a Recipe can move between spreadsheets, applications, organizations and the public web without being rewritten each time.

The result is a portable analytical substrate for knowledge: open, composable, inspectable and executable.