Newt, the third prototype

R. S. Doiel, rsdoiel@caltech.edu

Caltech Library, Digital Library Development

What is Newt?

- A rapid application develop tool
  - for applications that curate metadata
- Audience: Libraries, Archives, Galleries and Museums

Findings from Prototype 2:

> *Is Newt and "off the shelf" software enough to create metadata curation applications?*

Short answer is **yes**. Longer answer is more nuanced.

Findings from Prototype 2:

Is Newt and "off the shelf" software enough to create metadata curation applications?

1. Newt's YAML file can grow very large
2. Managing the YAML file can be done interactively
3. Model vetting and validation should happen early in the data pipeline
4. Postgres+PostgREST is a complex back end

Questions raised by Prototype 2:

- Where do I focus my simplification efforts?
- What is a “good enough” interface for managing the YAML file?
- Mustache templates language are too simple, what should replace it?

High level Concepts (remain the same)

- describe the application you want
- generate the application you described
- running the application using a service oriented architecture

Implementation Concepts (remaining the same)

- JSON data sources
- data modeled in YAML
- routing requests through data pipelines
- simple template engine renders JSON as HTML

Themes (remains the same)

- Pick Simple = (No coding) + (Less coding)
- Compose applications by combining models with data pipelines
- Avoid inventing new things

Goal of Prototype 3: Questions to explore

1. Is Handlebars a good fit for managing data views and rendering HTML?
2. Is generated TypeScript validation middleware the right fit for a validation?
3. Should Postgres+PostgREST remain the exclusive back end of Newt?
4. Should the generate step subsume the external Postres commands?
5. Should the generate step generate the validation middleware binary?

Changes from last prototype

- Removed some Go cli (e.g. ws, mustache, newtmustache)
- The action "init" was renamed "config", now an optional action
- The action "generate" was subsumed by "build"
- Renamed newtrouter to ndr (Newt Data Router)
- Added nte (Newt Template Engine), a Handlebars template engine

Changes from last prototype

- “oid” was renamed “identifier”
- Interactive modeler and configuration simplified
- Experimenting with Deno+TypeScript validation middleware

Off the shelf (no coding)

- JSON Data Source
  - Postgres + PostgREST
- newt, ndr, and nte
- Deno compiles TypeScript validation middleware

Assemble app from YAML (less coding)

- Data modeling via a interactive user interface
- Results is expressed in YAML

How do I think things will work?

1. Model your data interactively
2. Generate our application code
3. (Re)create database, run setup.sql and models.sql
4. Run and test using Newt command

Shell example

```
newt model app.yaml
newt generate app.yaml
dropdb --if-exists app; createdb app
psql -c app '\i setup.sql'; psql -c app '\i models.sql'
newt run app.yaml
open http://localhost:8010
```

Here's a demo of the process

FIXME: link to a record demonstration here

Third prototype Status

- A work in progress (continuing through 2024)
- Working towards a version 1.0 release hopefully in 2025
- Using internal applications as test bed

How much is built?

- ☒ Newt developer tool
- ☒ Router is implemented and working
- ☒ ~~Mustache template engine is working~~ (removed)
- ☒ Newt template engine (supporting Handlebars templates)
- ☐ Modeler (testing and refinement)
- ☐ Generator development (refactor, testing and refinement)

Insights from prototypes 1, 2 & 3

- “Off the shelf” is simpler
- An interactive UI for managing YAML is helpful
- A validition needs to happen early in the data pipeline

Insights from prototypes 1 & 2

- SQL turns people off, use a code generator
- Hand typing templates is a turn off, use a code generator
- Large YAML structures benefit from code generation
- Automatic “wiring up” routes and templates is helpful

What’s next?

- Build real applications using Newt
- Get feedback for refinement
- Fix bugs

Out of the box

- `newt` the Newt development tool
- `ndr` the Newt data router
- `nte` the Newt Template Engine

Unanswered Questions

- Who is in the target audience?
- What is the minimum knowledge required to use Newt effectively?
- What is the best human interface for Newt?

Someday, maybe ideas

- A visual programming or conversational user interface
- Direct SQLite 3 database support and integration could be much simpler than Postgres+PostgREST
- Web components for library, archive and museum metadata types
- A S3 protocol web service implementing file storage using OCFL
- Render Newt apps into a standalone binary application

Related resources

- Newt https://github.com/caltechlibrary/newt
- Handlebars programming languages support
- Dataset + datasetd https://github.com/caltechlibrary/dataset

Thank you!

- This Presentation
  - pdf: https://caltechlibrary.github.io/newt/presentation3/newt-p3.pdf
  - pptx: https://caltechlibrary.github.io/newt/presentation3/newt-p3.pptx
- Newt Documentation https://caltechlibrary.github.io/newt
- Source Code: https://github.com/caltechlibrary/newt
- Email: rsdoiel@caltech.edu