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 for applications with many data models
2. Model vetting and validation should happen early in the data pipeline, ideally as a generated program and browser side too
3. Postgres+PostgREST is a powerful combination but it'd be nice to have something simpler
4. Managing the YAML file should be done interactively

Questions raised by Prototype 2:

- Where do I focus my simplification efforts?
- How do I ensure that large YAML files remaining human manageable?
- Mustache template language is a little 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 Postgres+PostgREST the right JSON data source to focus on?
3. Is generated TypeScript middleware the right fit for a validation service?

Changes from last prototype

- Removed some Go cli (e.g. ws, mustache, newtmustache)
- The action “init” was renamed “config”, an optional action
- Renamed newtrouter to ndr (Newt Data Router)
- Added nte (Newt Template Engine) supporting Handlbars templates
- Generating Handlebars templates
- Generating TypeScript validator as middleware run via Deno
- oid was renamed “identifier” to clearity

Off the shelf (no coding)

- JSON Data Source
  - Postgres + PostgREST
- newt, ndr, and nte
- Deno to run generated 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. Run and tesst app

Steps one is interactive

```
newt model app.yaml
```

Step two, generate our code

```
newt generate app.yaml
```

Create Postgres+PostgREST setup and schema (e.g. SQL files) Generate Handlebars templates Creates a TypeScript model validation service Wires up routes and template mappings

Step three run and tet

```
newt run app.yaml
```

Point your web browser at http://localhost:8010 to test

Here’s a demo of the process

FIXME: link to a record demonstration here

Third prototype Status

- A work in progress (continuing through 2024)
- A Working version 1.0 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
- A validition service in TypeScript lets us leverage the same generated code in the browser
- An interactive UI for managing YAML is helpful

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” of routes and templates very helpful

What's next?

- Build some 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

- What is the minimum knowledge required to use Newt effectively?
- Who is in the target audience?
- Would a visual programming approach or conversation user interface make sense?

Someday, maybe ideas

- A visual programming or conversational approach could be easier for managing the YAML file
- 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 object storage using OCFL
- Generate code which can compile stack into a single binary application

Related resources

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

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