TANSU — A workflow for cabinet layout

Pavneet Arora

Abstract

A workflow using ConTEXt and Asymptote to design cabinet layouts and evaluate the impact of the design to costing is discussed. This work builds on the YAWN workflow which suggested the use of TEX as the "View" in a Model-View-Controller framework.

1 Introduction

At the TUG 2012 conference and in *TUGboat* 33:2 (2012), the YAWN workflow [1] was presented. It emphasized considering TEX as the "View" in a Model-View-Controller framework.

YAWN, however, focused on the steps leading up to the final document production. It concerned itself more with the *Model* and the *Controller*, the former utilizing YAML [2] and the latter utilizing Ruby, than with the layout of the resulting document, which was a simple text document akin to what might have come off a line printer.

This article extends the work of YAWN to take on the problem of cabinet layout, with a particular emphasis on the presentation of the final document. It does so while continuing to use the elements of YAWN as a design pattern.

It is natural for a client to seek alternative design layouts for storage units in areas such as kitchens, offices, and libraries even before awarding a project to a specific vendor. The layout is invaluable in visualising how the space might be used, and what options can fit in the allotted space. An adjunct to the layout is, of course, the question of cost. How much will a change in the layout cost?

The challenge for the prospective vendor, on the other hand, is to offer up sufficient detail to the client without over-committing resources to this exercise, since this is a fixed expense which may or may not be recouped later on. Shortening the time required to generate these layouts and associated costs affords a vendor two advantages: first, an opportunity to explore layout options alongside a customer, and second, a way to monitor the implications of the layout changes to the production cost, thus ensuring the viability of their own quotation.

Even when a layout is decided upon, it is useful to evaluate the offerings from different vendors against the design to compare costs. Again, as advocated in YAWN, decoupling the configuration from the catalogue of components from different manufacturers allows *rapid estimating*.

2 TANSU

In homage of the location of TUG 2013, Tokyo, the cabinet layout workflow is named TANSU — tansu (タンス) being the Japanese word for the traditional storage unit.

TANSU — the acronym — is derived from the following:

- TEX and
- Asymptote driven
- Nomenclature for
- Storage
- Unit layout.

3 Specification

The specification of the layout is a straightforward YAML file. Here is an example:

```
:projectID: 1923IMPHOT
:projectAddress: |
    Imperial Hotel Apartments
    Frank Lloyd Wright Edition (1923)
    Tokyo, Japan
:clientName: Okura Kihachiro
:cabinetSpec:
    :manufacturer: Fabritec
    :series: EuroStyle
    :walls:
        -
            :wall:
                :name: East
                :baseCabinets:
                    - HD30844D
                    - B2D24
                    - S24
                    - BSD30
                    - HD1584-R
                :wallCabinets:
                    - HD30844D
                    - W1230
                    - W2430
                    - W1230-R
                    - W3015HZ
                    - HD1584-R
```

The project in question is the fictitious Imperial Hotel Apartments. Baron Kihachiro, one of the investors in the Frank Lloyd Wright-designed Tokyo Imperial Hotel of 1923, has decided to develop a set of apartments along the same lines. However, there is no budget for the custom cabinets that Wright would no doubt insist upon, so we have been approached to come up with design options and their associated costs.

Each wall is given as an array of *base* cabinets (cabinets affixed to the ground) and *wall* cabinets

(cabinets mounted on a wall). The hybrid, *tall* cabinets which extend from floor to top, e.g., HD30844D and B1584-R, need to be listed in both arrays.

The notation assigned to each cabinet is taken here from the catalogue of the specified manufacturer, Fabritec, but it is generic enough to be applied to the offerings of other manufacturers. So, for instance, B2D24 indicates a 2-drawer cabinet that is nominally 24 inches wide, a common cabinet option.

In building catalogues for different manufacturers or even different cabinet series from the same manufacturer, the same notation, once decided upon, can be applied across all the catalogues.

Often, a single cabinet model number is used without an indication of door swing, which is switchable during the field installation. However, for the purpose of visualisation a suffix in the specification allows the layout to indicate intent as in W1230-R for a *right-hinged* door; the default door swing is taken to be *left-hinged*, as in the cabinet W1230 where no suffix is given.

4 Catalogue

A catalogue is similarly constructed using YAML, a sample cabinet from which is given here:

```
:cabinets:
    :subcategory: Cabinets
    :items:
        -
            :model: BD24
            :width: 24"
            :height: 30 1/4"
            :depth: 23 5/8"
            :doors:
                -
                    -
                        :swing: :drawer
                        :width: 24"
                        :height: 4"
                -
                    -
                        :swing: :left
                        :width: 12"
                        :height: 26 1/4"
                    -
                        :swing: :right
                        :width: 12"
                        :height: 26 1/4"
            :desc: 1-drawer 24"W base cabinet
            :price: 362.27
```

One thing that is immediately apparent is a need to deal effectively with *customary units* of length, i.e., feet-inches-fractional inches. Fortunately, there exists a succinct — and one I would consider beautiful and even poetic because of it — regular expression that does just that (with some slight modification for use in Ruby) [4] returning the matched parts:

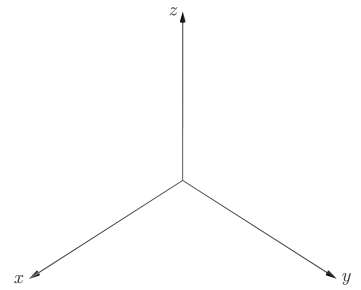
```
re=%r{(?:(?:(?<Feet>\d+)[ ]*(?:'|ft)){0,1}[ ]
*(?<Inches>\d*(?![\/\w])){0,1}(?:[ ,\-]){0,1}
(?<Fraction>(?<FracNum>\d*)\/(?<FracDem>\d*))
{0,1}(?<Decimal>\.\d*){0,1}(?:\x22| in))|
(?:(?<Feet>\d+)[ ]*(?:'|ft)[ ]*){1}}
```

The constituent parts of each cabinet are specified in the `:doors:` hash: they consist of rows of drawers and doors represented as a two-dimensional array. In TEX terms, they may be thought of as cells in a table, some of which span columns (only). So in this case, the first row from the top is a single drawer 24-inches wide by 4-inches high. The next row consists of two doors hinged to the outside frame swinging out from the middle.

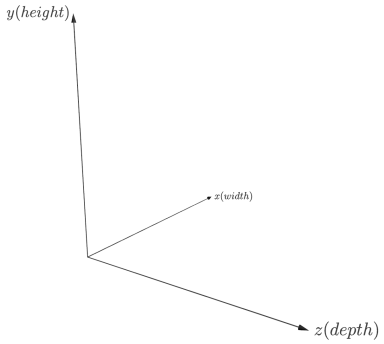
5 Output

The *Model* consists of the aforementioned *specification* and *catalogue*, while the controller is implemented in Ruby to analyse the two.

TANSU builds up a wall layout by retrieving the catalogue entry for each cabinet in the specification, then mapping the cabinet dimensions in the coordinate space of Asymptote [3]. The native coordinate system for Asymptote is the traditional mathematical one (the difference in apparent label size is due to 3D perspective):



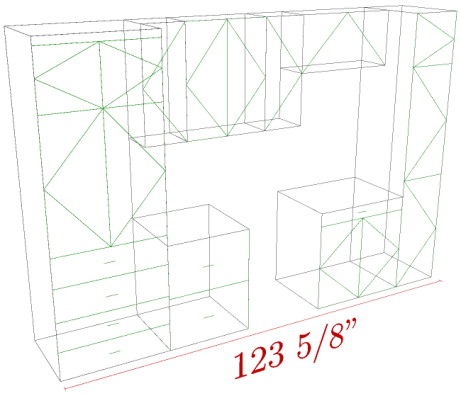
However, for cabinet layout the natural coordinate system is the following:



Once the cabinet box is drawn, doors are overlaid and their swings indicated.

What then is the output? Well, a distinct advantage of using Asymptote is that not only can it handle three-dimensional drawing with aplomb and produce EPS output from the drawing instructions, but it also has a built-in OpenGL renderer allowing one to perform transformations interactively, such as rotations and changes to the viewport. This makes it simple to explore the generated layout from different angles. Here is the rendered layout for the above specification:

Client Name ID:	Okura Kihachiro
Project ID:	1923IMPHOT
Project Address:	Imperial Hotel Apartments Frank Lloyd Wright Edition (1923) Tokyo, Japan



The associated costs for this layout are typeset using ConTEXt's *Natural Tables* mechanism [5] and shown in the report as follows:

Base Cabinets

Model No.	Description	Price
HD30844D	30" x 84" tall cabinet with 4 drawers	$1,192.89
B2D24	2-pot drawer 24"W base cabinet	$362.98
S24	24"W base cabinet opening	$250.64
BSD30	24"W sink cabinet with drawer face	$344.77
HD1584	15" x 84" tall cabinet	$501.75
	Sub-total	$2,653.03

Wall Cabinets

Model No.	Description	Price
W1230	12" wall cabinet	$136.13
W2430	24" wall cabinet with glass door	$254.80
W1230	12" wall cabinet	$136.13
W3015HZ	24" x 15" bridge cabinet, top hinge	$176.93
	Sub-total	$703.99

$3,357.02

TANSU demonstrates that the ideas described in YAWN, namely of considering TEX as the *View* in an M-V-C framework and representing the *Model* in YAML, offers an extensible methodology with wide-ranging applications. In a completely different domain from that used to initially demonstrate YAWN the same workflow has been used with success.

References

[1] Pavneet Arora. YAWN — A TEX-enabled worflow for project estimation. http://tug.org/TUGboat/tb33-2/tb104arora.pdf, 2012.

[2] Oren Ben-Kiki, Clark Evans, and Ingy. YAML Ain't Markup Language (YAML) version 1.2. http://www.yaml.org/spec, October 2009.

[3] John Bowman et al. Asymptote: The vector graphics language. http://asymptote.sourceforge.net.

[4] Normand Frechette. Feet-inch to Decimal. http://regexlib.com/REDetails.aspx?regexp_id=2127.

[5] Hans Hagen et al. ConTEXt natural tables. http://wiki.contextgarden.net/TABLE.

◇ Pavneet Arora
pavneet_arora (at)
bespokespaces dot com
http://blog.bansisworld.org