XQuery Web Frameworks

(: IMHO / Rant :)

adam@exist-db.org

existsolutions
www.existsolutions.com

What is an XQuery Web Framework?

- What is a (good) Web Framework?
  - Typically an API and/or Container
  - Reduces boilerplate code
  - Clean - you do NOT have to change your coding style
    (POJO → POXQM)
  - Forces separation of Business Logic from UI
  - URL Mapping/Rewriting
- Goal: Build re-useable XQuery Web Apps
- Why?
  - XQuery bridges the impedance mismatch between code and data. Work in one environment!
  - XQuery *is* suitable for enterprise scale application development
  - XQuery + Web = rapid development (Agile prototypes)

What's already available?

- XQMVC
  - Heavy-weight, full MVC.
  - Written in XQuery 1.0 + vendor extensions.
  - Runs on MarkLogic and eXist-db (almost).
  - Impressive! Build by convention. Almost transparent...
  - URL Rewriting Support.
- Mustache
  - Light-weight, just VC.
  - Written in XQuery 1.0 (+3.0 try/catch) + vendor extensions.
  - Runs of MarkLogic and eXist-db (almost?).
  - View uses JSON for template Syntax (Ughh?!?)
  - Fun. Easy. Its an invokable function library. Not transparent!

Every time I...

~ The same old ~

Start building an XQuery Web Application...

- I dont like littering my XQuery with XHTML (or presentation XML)
  - As it tightly couples the look and feel with my code.
  - XQMVC is too heavy, Mustache is too light – tough!
  - So, I abstract. Building a bespoke templating framework (XSLT or XQuery) to merge the output of my business logic with my view.
- I worry about RESTful and Cool (descriptive+persistent) URI's
  - eXist-db and others have URI rewriting routers that you can implement in XQuery. Useful, but IMHO - yuck! Its not declarative or easy to understand intention. Easy to become muddled.
  - Rewrite project looks interesting, its strictly declarative. Limited to MarkLogic only at the moment. I think we can do better...
- Re-invent ~70% of my already square wheel. Not reusable? / Standards?

I have a dream...

Please let it be...

Standards (EXPath?) - two independent things to solve:

1) De-Coupling view from business logic → Templating.
    1) KISS! Merge the output of an XQuery/XSLT with an XHTML/XML Template. Just use @id or @xml:id.
    2) Where the id's match you merge (recursively?)
    3) HTML Tables are hard (impossible?) to merge (without re-creating/coupling on both sides)... Mustache actually makes sense here!
2) URI Routing/Rewritting
    1) Must be declarative and Intuitive. LiftWeb interesting, but not declarative.
    2) Simples! - Use XQuery 3.0 Annotations on module functions, web requests can be routed directly to your functions. (e.g. Java Jersey for REST)

I have a dream...

Why not this?

RESTFul XQuery 3.0 Annotations

```
declare
%rest:listen("GET", "/sausages/list")
function local:list-sausages() {
    <sausages>
      { ... }
    </sausages>
};

declare
%rest:listen("GET", "/sausage/{$id}")
function local:get-sausage($id as xs:string) {
    <sausage id="{$id}">
    { ... }
    </sausage>
};
```

Nice, right???