

Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



# Reflections on making XQMVC platform agnostic

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



*...this is just a collection of excited thoughts,  
and not a well thought out presentation!*

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## The Story so far -

- General Idea – Comparison
- Take an existing XQuery Application running on MarkLogic and run it on eXist-db

*“Easy”, ...right?!?*

- Looking at the code, we discover it is written on top of the XQMVC framework
- XQMVC is Open Source, but MarkLogic specific – Lets port for eXist-db

*...or, go a little further and make XQMVC platform agnostic :-)*

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## XQMvc Requirements

- XQuery (was Xquery-1.0-ml)
- HTTP Context – its the glue!
- Documents - Storage, Retrieval and Removal (was fn:doc and xdmp: extensions)
- Collections (was xdmp:directory, not sure why it wasnt fn:collection)
- Node Level Updates (was xdmp: extensions)
- Optional URL Rewriting (was MarkLogic specific)  
e.g. /index.xql?\_c=welcome&\_f=index -> /welcome/index

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## How to make platform agnostic?

### 1. EXPath and EXQuery

- Ideal scenario – Wraps implementation specifics in standardised functions.
- Not ready today, and also not widely implemented yet.

### 2. Strategy Pattern (from XQuery Design Patterns by 28msec Inc.)

- Requires XQuery 1.1 for HoF's (Not widely implemented).
- Could of used processor specific HoF's, but defeats the point!

### 3. Interface with Processor Specific Modules

- Generic central wrapper module acts as an Interface to a Processor.
- One Module for each supported Processor. Contains non-standard code.

# Interface with Processor Specific Modules

```
module namespace xqmvc =  
  "http://scholarsportal.info/xqmvc/core";  
  
import module namespace processor =  
  "http://scholarsportal.info/xqmvc/system/processor"  
  at "processor/processor.xqy";  
  
processor:execute-module-function(  
  xqmvc:get-namespace-for-prefix("xqmvc-ctrlr"),  
  xs:anyURI($controller-file),  
  $function)
```

xqmvc.xqy

processor.xqy

```
module namespace impl =  
  "http://scholarsportal.info/xqmvc/system/  
  processor/impl/exist-db";  
  
declare function impl:execute-module-function(  
  $module-namespace as xs:anyURI,  
  $controller-file as xs:anyURI,  
  $function-name as xs:string) as item()* {  
  
  (: eXist-db processor specifics i.e. util:eval(...) :)  
  
};
```

```
module namespace processor =  
  "http://scholarsportal.info/xqmvc/system/processor";  
  
import module namespace xqmvc-conf =  
  "http://scholarsportal.info/xqmvc/config"  
  at "../application/config/config.xqy";  
  
(: choose a processor :)  
import module namespace impl =  
  "http://scholarsportal.info/xqmvc/system/processor/impl/exist-db"  
  at "impl/exist-db/impl.xqy";  
  
(: import module namespace impl =  
  "http://scholarsportal.info/xqmvc/system/processor/impl/marklogic"  
  at "impl/marklogic/impl.xqy"; :)  
  
declare function processor:execute-module-function(  
  $module-namespace as xs:anyURI,  
  $controller-file as xs:anyURI,  
  $function-name as xs:string) as item()* {  
  
  impl:execute-module-function(  
    $module-namespace, $controller-file, $function-name)  
  
};
```

```
module namespace impl =  
  "http://scholarsportal.info/xqmvc/system/  
  processor/impl/marklogic";  
  
declare function impl:execute-module-function(  
  $module-namespace as xs:anyURI,  
  $controller-file as xs:anyURI,  
  $function-name as xs:string) as item()* {  
  
  (: MarkLogic processor specifics i.e. xdmp:eval(...) :)  
  
};
```

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## Issues along the way -

### 1. Processor specific extension functions

- All processors have different yet similar functions! EXPath aims to solve this.
- `xdmp:set(...)` can be replaced with XQuery 1.0 FLOWR or recursion.
- `xdmp:invoke(...)` and `xdmp:eval(...)` are abstracted through `processor.xqy`

### 2. XQuery 1.0-ml

- Can be rewritten as XQuery 1.0 for portability (or abstracted).
- Beware of relaxed namespace rules around module functions and variables
  - Find XQST0045 errors and fix.

### 3. Processor specific types (e.g. `map:map`)

- Similar to functions can be rewritten using standard datatypes (or abstracted).

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## Issues along the way -

### 4. Document Updates

- `xdmp:node-*` can be abstracted through `processor.xqy`
- Ideally should be XQuery Update Facility 1.0, but not supported by stakeholder processors. Standard has yet to be finalised!



# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## Reflections -

### 1. XQuery 1.0 is not enough

- Additional processor functions needed – e.g. http, and eval.
- EXPath will address this in time.
- Best approach today is to abstract and contain processor specifics?

### 2. XQuery 1.1 will still not be enough!

- HoF's will remove almost all uses of eval (enabling better SoC).
- Support for dynamically loading modules and calling their functions is missing!
  - Will still require some sort of `evil` eval :-)

# Adam Retter

<!-- Independent Consultant / eXist-db Developer -->



## Status -

- Almost finished. Just optional URL Rewriting remaining.
  - Code is available - <https://xqmvc.googlecode.com/svn/branches/diversify/>
  - Next release of XQMVC will be platform agnostic.
- 
- XQuery Implementers – why not create a module for your processor?
    - Maybe just one hours work!

## Questions?