Adventures with BaseX and web applications

Andy Bunce @apb1704

Feb 2013

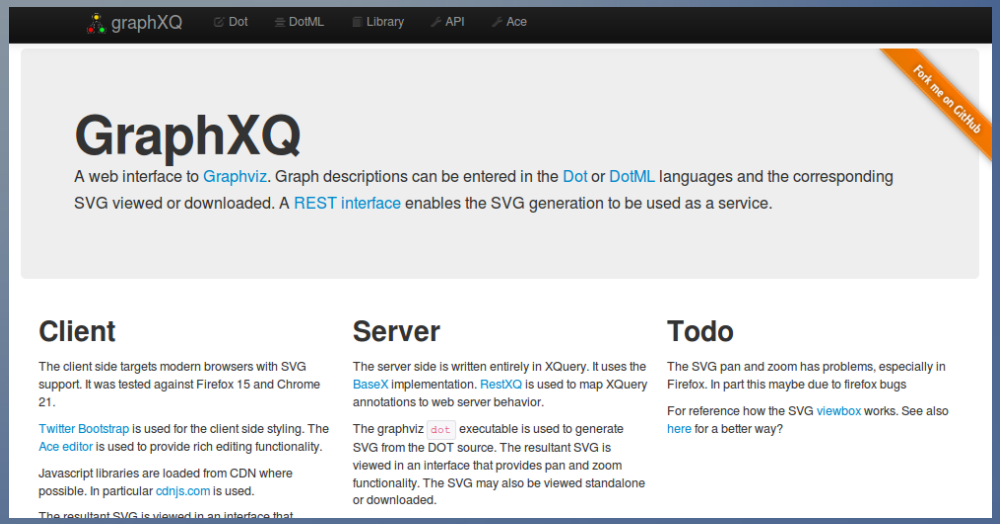
BaseX and the Web

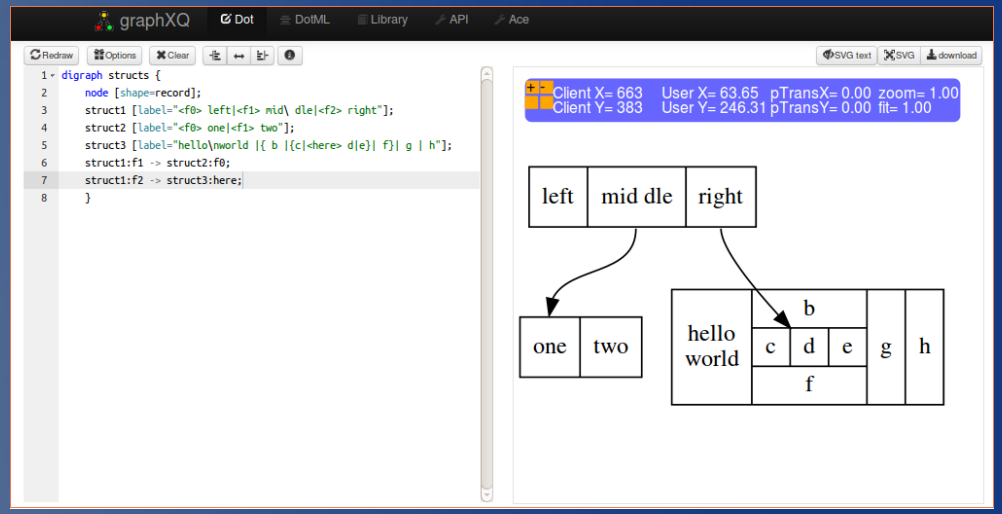
- GraphXQ – RESTXQ + Graphviz
- (Using a PaaS)
- CellarXQ – Angular.js + OAuth
- BaseX with Node.js + events

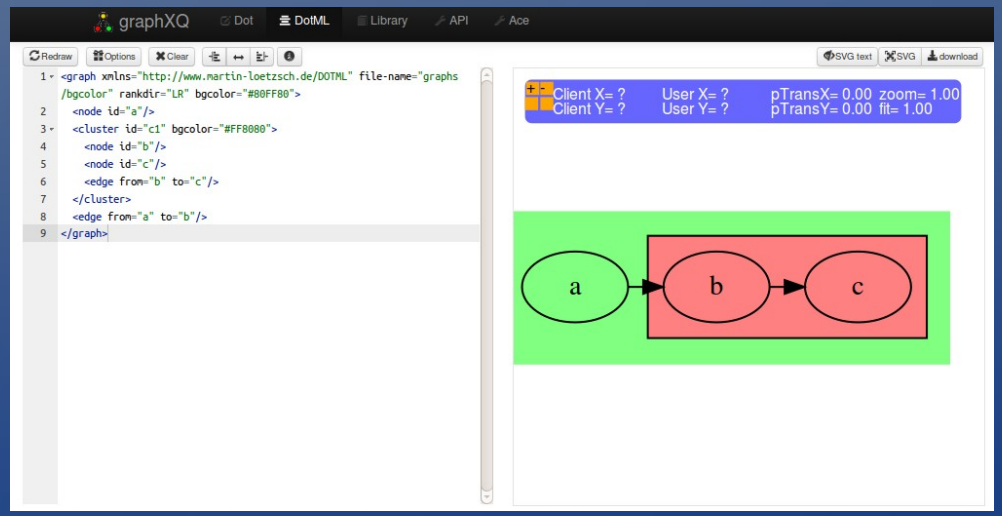
RESTXQ http://docs.basex.org/wiki/RESTXQ

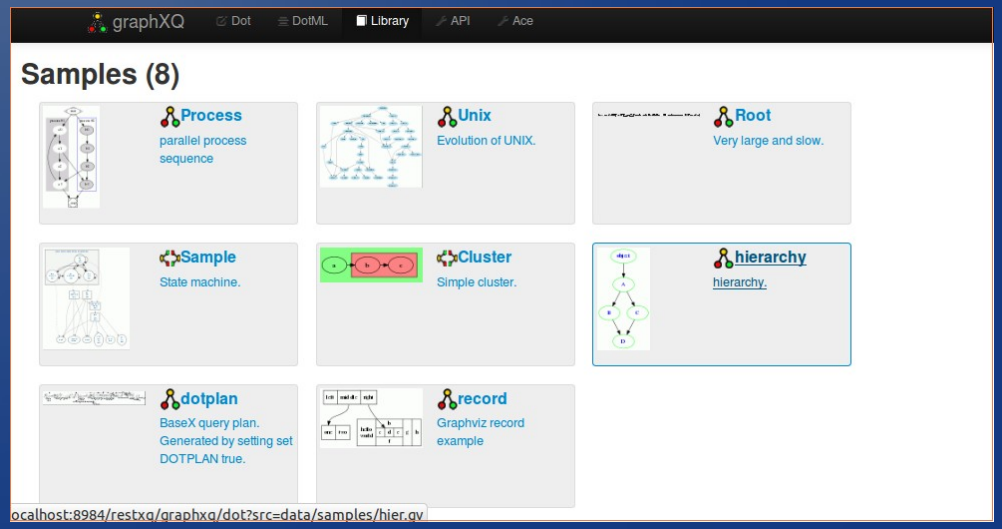
- introduced by Adam Retter, is an API that facilitates the use of XQuery as a Server Side processing language for the Web.
- inspired by Java's JAX-RS API.
- a set of XQuery 3.0 annotations for mapping HTTP requests to XQuery functions.
- the XQuery functions generate and return HTTP responses.

GraphXQ – RESTXQ + Graphviz

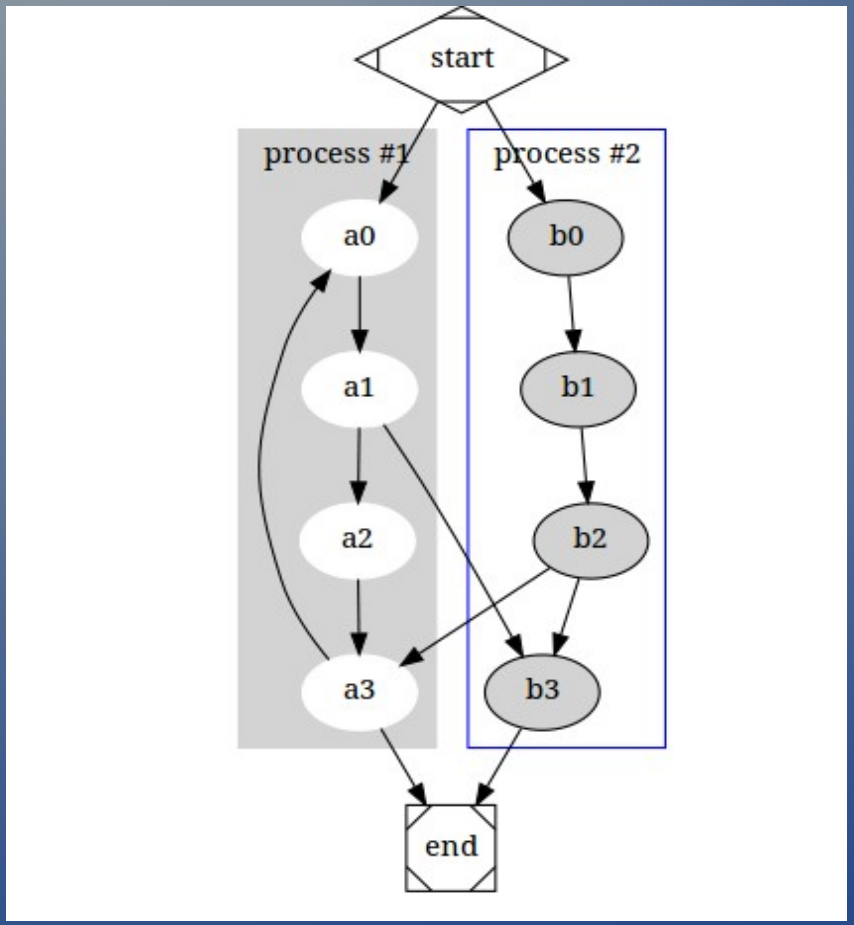








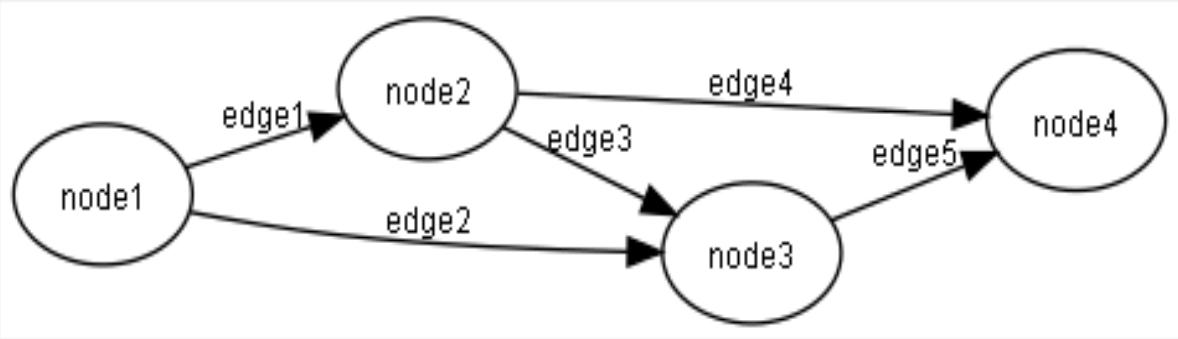
Graphviz http://www.graphviz.org/



- Open source graph (network) visualization project from AT&T Research.
- directed and undirected graph layout
- i/p is string (dot syntax)
- o/p is SVG

DOTML http://martin-loetzsch.de/DOTML/

- DotML is a XML based syntax for the input language of the 'Dot' graph drawing tool




```
<graph file-name="graphs/nice_graph" rankdir="LR">
  <node id="a" label="node1" fontsize="9" fontname="Arial"/>
  <node id="b" label="node2" fontsize="9" fontname="Arial"/>
  <node id="c" label="node3" fontsize="9" fontname="Arial"/>
  <node id="d" label="node4" fontsize="9" fontname="Arial"/>
  <edge from="a" to="b" fontname="Arial" fontsize="9" label="edge1"/>
  <edge from="a" to="c" fontname="Arial" fontsize="9" label="edge2"/>
  <edge from="b" to="c" fontname="Arial" fontsize="9" label="edge3"/>
  <edge from="b" to="d" fontname="Arial" fontsize="9" label="edge4"/>
  <edge from="c" to="d" fontname="Arial" fontsize="9" label="edge5"/>
</graph>
```

Graphviz from BaseX

- Use proc:execute()
- Sadly has no pipe in so use temp file

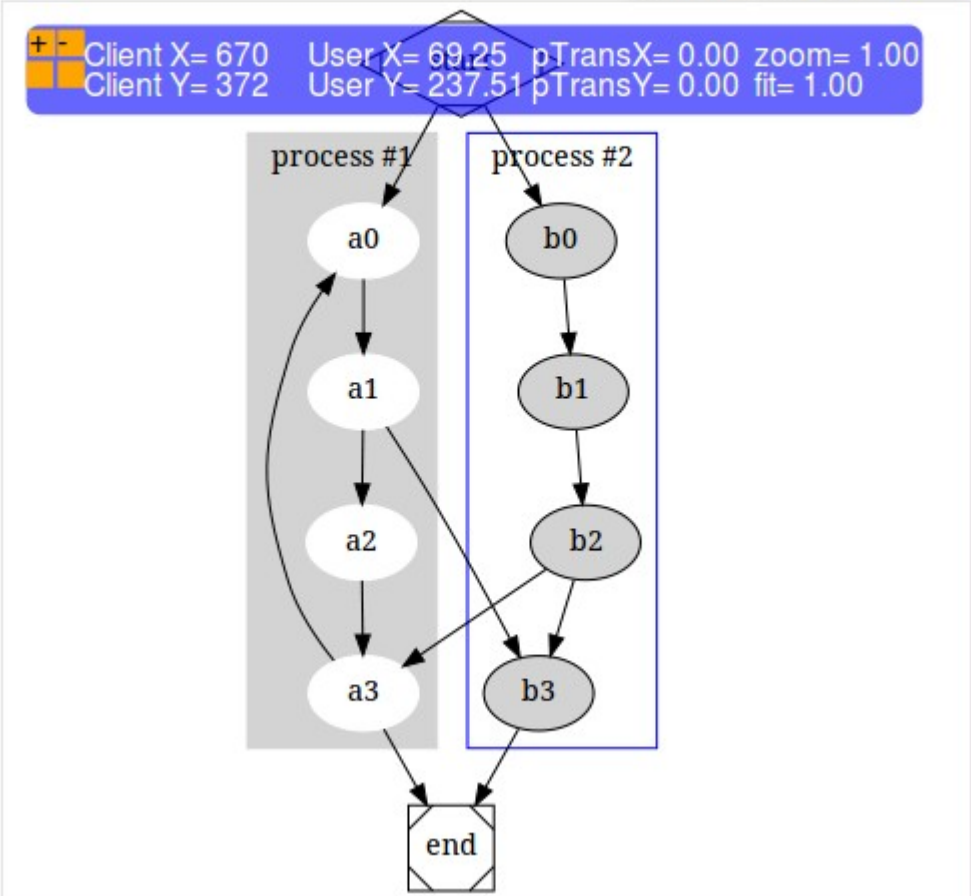
```
declare %private function dot1( $dot as xs:string) as element(svg:svg)
{
    let $fname:=$gr:tmpdir || random:uuid()
    let $junk:=file:write-text($fname,$dot)
    let $r:=proc:execute($gr:dotpath , ("-Tsvg",$fname))
    let $junk:=file:delete($fname)
   (: let $r:=fn:trace($r,"hhi"):)
    return if($r/code!="0")
           then  fn:error(xs:QName('gr:dot1'),$r/error)
           else (: o/p  has comment nodes :)
                       let $s:=fn:parse-xml($r/output)
                let $ver:=$s/comment()[1]/fn:normalize-space()
                                let $title:=$s/comment()[2]/fn:normalize-space()
                let $svg:=$s/*
                return   <svg xmlns="http://www.w3.org/2000/svg"
                xmlns:xlink="http://www.w3.org/1999/xlink" >
  {$svg/@* ,
```

GraphXQ

graphXQ | Dot | DotML | Library | API | Ace

Redraw | Options | Clear | SVG text | SVG | download

```
1  digraph process {
2
3      subgraph cluster_0 {
4          style=filled;
5          color=lightgrey;
6          node [style=filled,color=white];
7          a0 -> a1 -> a2 -> a3;
8          label = "process #1";
9      }
10
11     subgraph cluster_1 {
12         node [style=filled];
13         b0 -> b1 -> b2 -> b3;
14         label = "process #2";
15         color=blue
16     }
17     start -> a0;
18     start -> b0;
19     a1 -> b3;
20     b2 -> a3;
21     a3 -> a0;
22     a3 -> end;
23     b3 -> end;
24
25     start [shape=Mdiamond];
```



Twitter bootstrap

http://twitter.github.com/bootstrap/

Home | Get started | Scaffolding | Base CSS | Components | JavaScript | Customize | Bootstrap

- Dropdowns
- Button groups
- Button dropdowns
- Navs
- Navbar
- Breadcrumbs
- Pagination
- Labels and badges
- Typography
- Thumbnails
- Alerts
- Progress bars
- Media object
- Misc

Navbar

Basic navbar

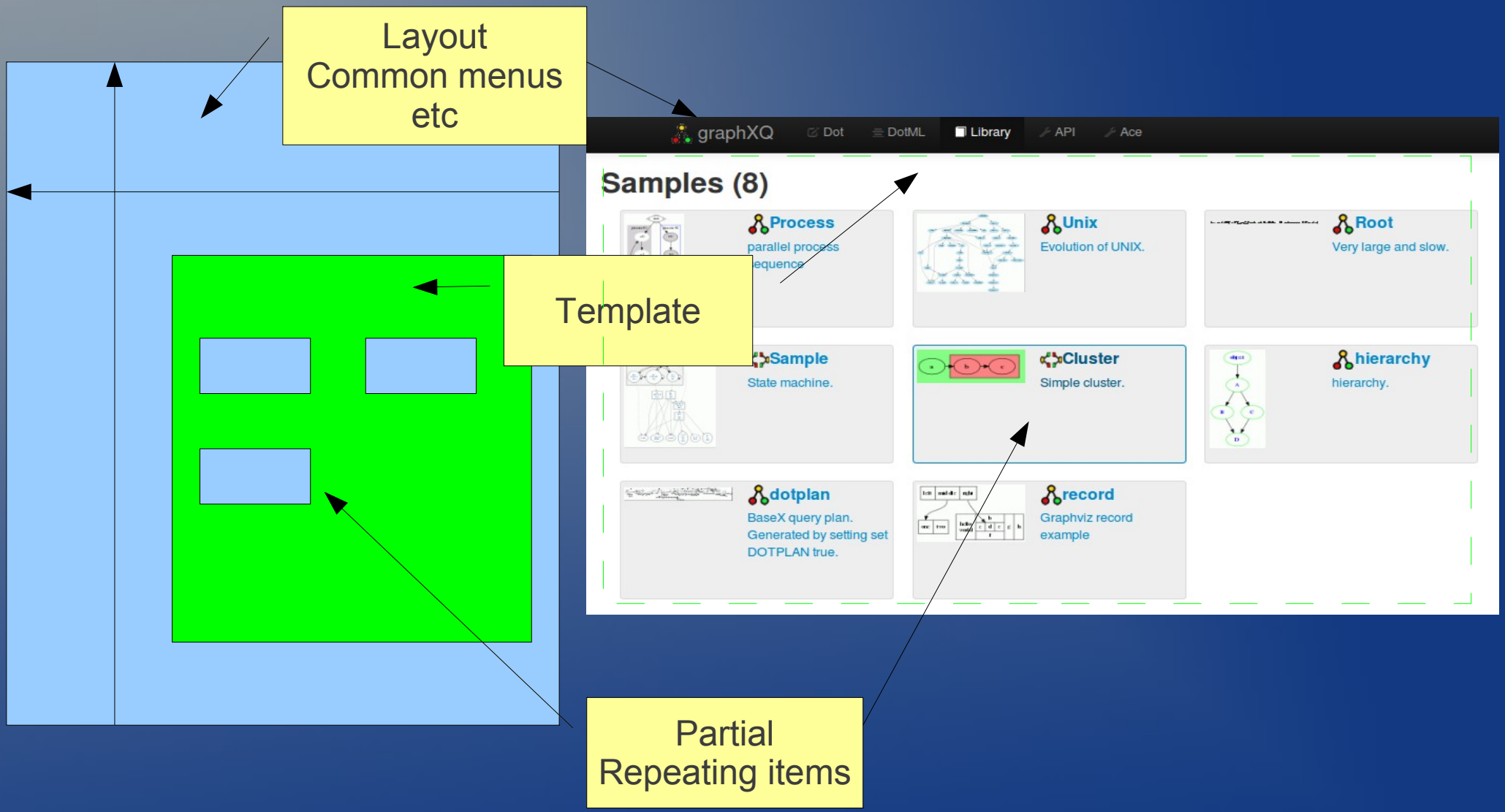
To start, navbars are static (not fixed to the top) and include support for a project name and basic navigation. Place one anywhere within a `.container`, which sets the width of your site and content.

Example

Title | Home | Link | Link

```
1.  <div class="navbar">
2.    <div class="navbar-inner">
3.      <a class="brand" href="#">Title</a>
4.      <ul class="nav">
5.        <li class="active"><a href="#">Home</a></li>
6.        <li><a href="#">Link</a></li>
7.        <li><a href="#">Link</a></li>
8.      </ul>
9.    </div>
10. </div>
```

Web page structure



TXQ - Templating

http://cubeb.blogspot.com/2012/11/xquery-templating-engines-and-txq.html

```
(:~
: template function
: @param template url to fill
: @param map name and value to apply
: @return updated doc from map
:)
declare function render($template as xs:string,$map as map(*)){
   let $map:=map:new(($map,map{"partial":=partial(?,?,?,$map,$template)}))
   return xquery:invoke($template,$map)
};

(:~
: template function with wrapping layout
: @param layout
: @return updated doc from map
:)
declare function render($template as xs:string,$map as map(*),$layout as xs:string){
    let $content:=render($template,$map)
    let $map:=map:new(($map,map{"body":=$content}))
    return render($layout,$map)
};

(:~
: partial template function: evaluate part for each value in sequence
: @return updated doc from map
:)
declare function partial($part as xs:string,$name,$seq,$map,$base){
  for $s in $seq
  let $map:=map:new(($map,map{$name:=$s}))
  return render(fn:resolve-uri($part,$base),$map)
};
```

Templating in use

```
declare
%restxq:GET %restxq:path("graphxq/library")
%output:method("html") %output:version("5.0")
function library(){
 let $lib:=fn:doc("data/library.xml")
 let $map:=map{"title":="Samples",
               "items":=$lib//items,
               "url":=function($item){fn:concat($item/url/@type,'?src=data/samples/',$item/url)}
               }
 return render("views/library.xml",$map)
};
```

library.xml

```
<div class="row-fluid">
     <h2>Samples ({fn:count($items/item )})</h2>
     <ul class=" thumbnails media-list">
     {$partial("item1.xml","item",$items/item)}
     </ul>
</div>
```

```
(:~
: Render html page
: @param template path to page template
: @params locals map of page variables
:)
declare function render($template as xs:string,$locals){
    let $path:=request:path()
    let $default:=map{ "title":=request:path(),
                       "active-link":=active-link($path,?), (: *** FAILS IF request:path()
                       "bodyclass":=""}
    let $locals:=map:new(($default,$locals))
    return txq:render(fn:resolve-uri($template),$locals,$grxq:layou
};
```

Application
Default values

Item1.xml

```
<li class="media thumbnail" style="height:128px;width:20em;">
    <a class="pull-left " href="{$url($item)}">
        <img class="media-object" src="/graphxq/data/thumbs/{$iten
       </a>
    <div class="media-body">
        <h4 class="media-heading">
           <a href="{$url($item)}">
           <img  src="/graphxq/{$item/url/@type}.png" />
           {$item/title/fn:string()}</a>
        </h4>
     {$item/description/node()}
    </div>
</li>
```

Using a PaaS

BaseX in the Cloud

Openshift https://openshift.redhat.com/app/

OPENSHIFT

TAKE YOUR APPS TO THE CLOUD

OpenShift is a next generation application hosting environment. Read on to see how it can help you achieve your application goals.

SIGN UP TO TRY **OPENSHIFT**

LEARN MORE

- Overview
- Pricing
- Enterprise PaaS
  - Challenges for Enterprises
  - Features and Benefits
- Get Started
  - Java

OpenShift Platform as a Service (PaaS)

OpenShift is Red Hat's Cloud Computing Platform as a Service (PaaS) offering. OpenShift is an application platform in the cloud where application developers and teams can build, test, deploy, and run their applications.

OpenShift takes care of all the infrastructure, middleware, and management and allows the developer to

Openshift features

- Has Java (and Node.js) pre-installed
- Has a free option
- Provides ssh terminal access

(:~

: There are other similar services

: E.g. https://www.appfog.com/

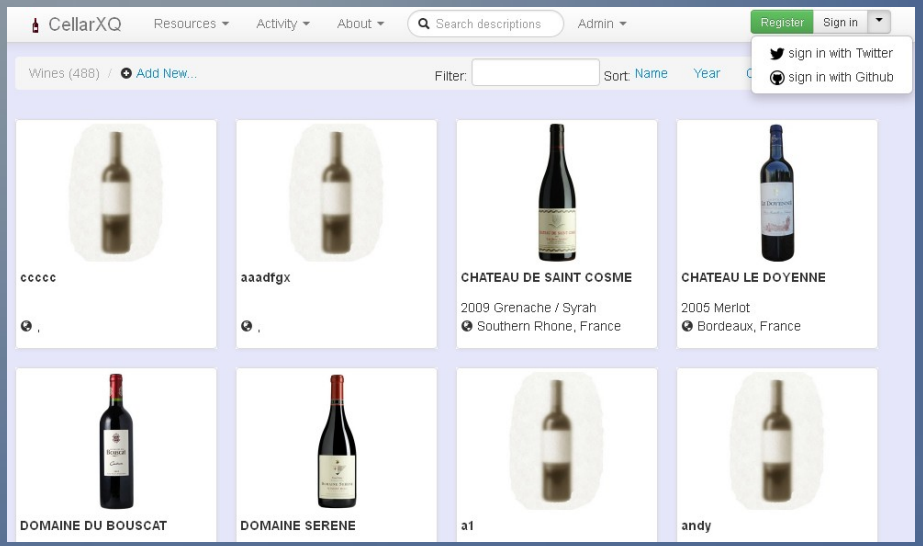
:)

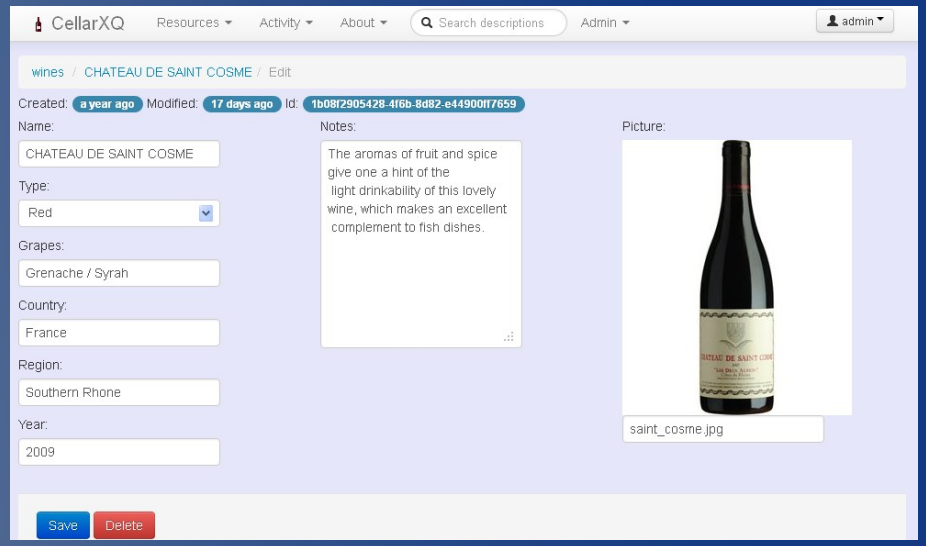
Openshift and BaseX

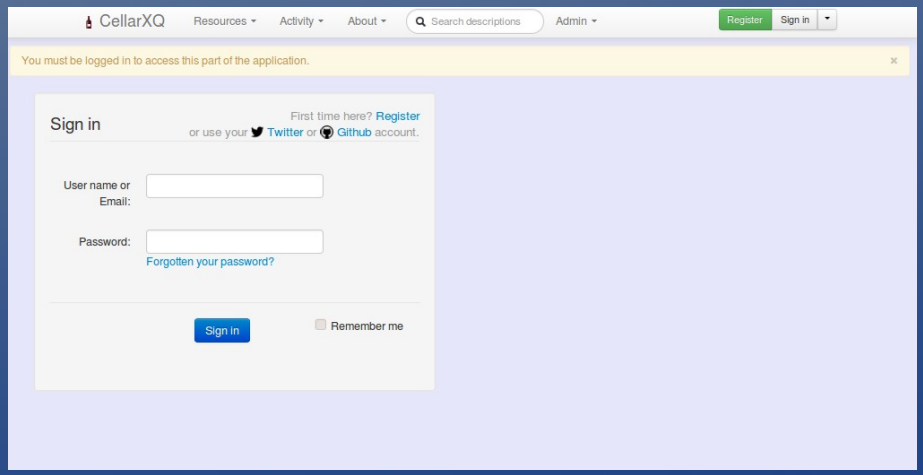
- Sign up, install tools
- Create a project using “DIY” cartridge
- This generates a git project with script stubs
- Add your code and deploy using `git push`
- Restrictions on usable ports

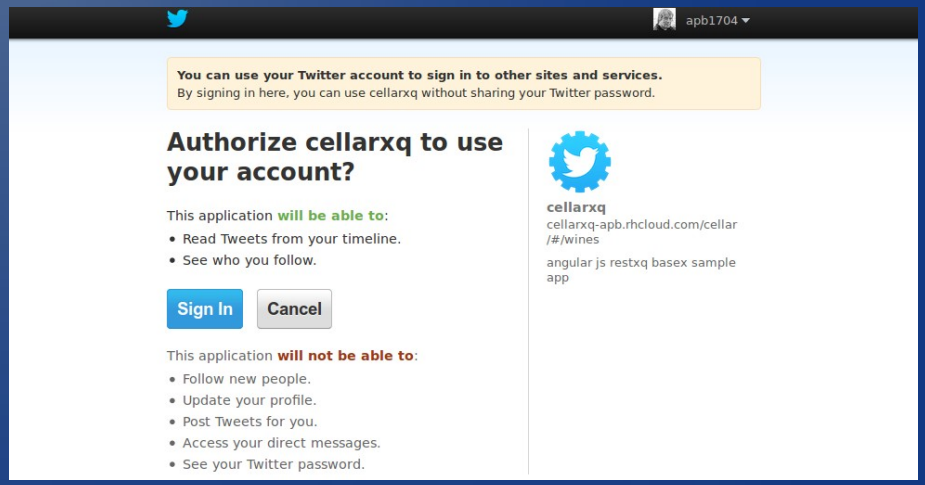
```
# start basex
cd ${OPENSHIFT_DATA_DIR}basex
nohup bin/basexhttp -X
-Dorg.basex.SERVERHOST=$OPENSHIFT_INTERNAL_IP \
-X -Dorg.basex.SERVERPORT=$OPENSHIFT_INTERNAL_PORT \
-p 15005 -e 15006 -s 15007 -U admin -P admin &
```

CellarXQ – Angular.js + OAuth









https://github.com/apb2006/basex-cellar

Based on http://coenraets.org/blog/2012/02/sample-application-with-angular-js/

angular.js http://angularjs.org/

- AngularJS is an open-source JavaScript framework from Google.
- Its goal is to augment browser-based applications with Model–View–Controller (MVC) capability.
- Declarative data binding
- Same space as XFORMS

Angular.js Hello world

The Basics

index.html

```
1.  <!doctype html>
2.  <html ng-app>
3.    <head>
4.      <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.0.4/
    angular.min.js"></script>
5.    </head>
6.    <body>
7.      <div>
8.        <label>Name:</label>
9.        <input type="text" ng-model="yourName" placeholder="Enter a name here">
10.       <hr>
11.       <h1>Hello {{yourName}}!</h1>
12.     </div>
13.   </body>
14. </html>
```

Hint: hover over `me`.

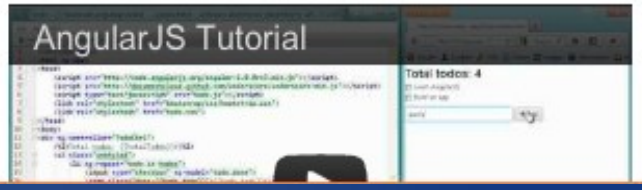
Edit Me

Name:

BaseX Prague

Hello BaseX Prague!

Watch as we build this app



JSON

- Angular expects to send and receive JSON
- The BaseX JSON serialization worked for me

But

- I had full control of the XML
- No namespaces etc...

The XML database

```
 </wine>
 <wine id="7000e620-f632-4b7f-bc73-f91a3bf8a07f">
   <meta created="2012-01-01T22:33:19.111Z" modified="2012-10-30T21:01:16.877Z"/>
   <name>DOMAINE SERENE</name>
   <year>2007</year>
   <grapes>Pinot Noir</grapes>
   <country>USA</country>
   <region>Oregon</region>
   <description>Though subtle in its complexities, this wine is sure to
please a wide range of enthusiasts. Notes of pomegranate will delight
as the nutty finish completes the picture of a fine sipping
experience.</description>
   <picture>domaine_serene.jpg</picture>
 </wine>
 <wine id="13cf6cdc-9c33-4a9c-a340-e8303c88d42e">
   <meta created="2012-01-01T22:33:19.111Z" modified="
   <name>LAN RIOJA CRIANZA</name>
```

```
(:~
: return name and id for all wines as json
:)
declare
%rest:GET %rest:path("cellar/api/wines")
%output:method("json")
function wines()
{
  <json arrays="json" objects="wine">
    {for $wine in $cellar:wines/wine
    order by fn:upper-case($wine/name)
    return <wine>{(
       meta-db:output($wine),
       $wine/name,
       $wine/year,
       $wine/grapes,
       $wine/region,
       $wine/country,
       $wine/picture
       )}</wine>}
  </json>
};
```

Updates

- Use db:output to mix updates and result

```
(:~
: update details for wine with id
: @modified timestamp used to detect lost update errors
:)
declare
%rest:PUT("{$body}") %rest:path("cellar/api/wines/{$id}")
%output:method("json")
updating function put-wine(
  $id,
  $body)
{
  let $old:=$cellar:wines/wine[@id=$id]
  return if($old) then
           let $items:=$body/json
           let $new:=  <wine id="{$old/@id}">
                        <meta created="{$old/meta/@created}"
                        modified="{fn:current-dateTime()}"/>
                        {$items/* except ($items/modified, $items/created,$items/id)}
                        </wine>
           return
               if($items/modified=$old/meta/@modified/fn:string() or fn:not($old/meta/@modified))
               then (replace node $old with $new,
                     events:log2("wine-mod",$id),
                     db:output($body)
                     )
               else db:output( web:status(403,"data modified"))
         else
            db:output(web:status(404,"Not found: " || $id))
};
```

The user database

- Users defined at the application level.
- Data in users.xml in the database


```xml
<users nextid="6">
  <user id="0">
    <name>guest</name>
    <avatar>guest.png</avatar>
    <data>
      <ace theme="dawn"/>
    </data>
  </user>
  <user id="1">
    <name>admin</name>
    <role>admin</role>
    <status>active</status>
    <avatar>admin.png</avatar>
    <auth type="local">oa2xJp0I39IG1DBdfa4Nzg==</auth>
    <stats created="2012-08-06T22:29:37.643+01:00" last="2013-01-21T14:41:30.796Z" logins="19"/>
    <data>
      <ace theme="dawn"/>
    </data>
  </user>
  <user id="2">
    <name>apb1704</name>
    <role>user</role>
    <auth type="twitter">apb1704</auth>
    <stats created="2012-12-06T11:17:14.082Z" last="2013-01-14T11:56:52.716Z" logins="11"/>
    <data>
      <ace theme="dawn"/>
    </data>
```

Sign in and user session

```
(:~
: login : will set userid into session if ok
: @return json indicating success or fail
:)
declare
%rest:path("cellar/auth/login")
%rest:POST("{$body}")
%output:method("json")
updating function login-post(
    $body)
{
 let $json:=$body/json
 let $u:=users:password-check($auth:userdb,$json/username,$json/password)
 return
     if($u) then
         (
           users:update-stats($auth:userdb,$u/@id),
           events:log2("login","local",$u),
            db:output((
                session:set("uid", $u),
                session-user($u) ))
                )
     else
      db:output(session-user(()) )
};
```

Register new user

```
(:~
: insert new user created with generate
:)
declare updating function create($userDb,$u as element(user))
{
    insert node $u into $userDb/users ,incr-id($userDb)
};

(:~
: increment the file id
:)
declare updating function incr-id($userDb)
{
     replace value of node $userDb/users/@nextid with next-id($userDb)+1
};
```

Permissions

- XQuery RESTXQ calls return status 404 if not permissioned.
- Page routing and login prompt is handled at the Angular level.

```
config(
        [ '$routeProvider', function($routeProvider) {

            $routeProvider.when('/wines', {
                templateUrl : 'partials/wine-thumbs.xml',controller: "WineListCtrl",
                reloadOnSearch: false
            })
            .when('/',{redirectTo : '/wines'})
            .when('/wines/add', {
               templateUrl : 'partials/wine-edit.xml',controller : "WineDetailCtrl" ,permission:".*"
            }).when('/wines/:wineId', {
                templateUrl : 'partials/wine-view.xml',controller : "WineDetailCtrl"
            }).when('/wines/:wineId/edit', {
                templateUrl : 'partials/wine-edit.xml',controller : "WineDetailCtrl",permission:".*"
            }).when('/grapes', {
                templateUrl : 'partials/grape-list.xml',controller : "GrapeListCtrl"
```

Application Configuration

- Stored in an XML file in WEB-INF
- Includes OAuth secrets and tokens

cellar-config.xml

```
<config>
<libserver>/lib</libserver>
   <aceserver>/lib/ace/486a740/src-min-noconflict</
   <twitter>
      <CONSUMER-KEY>XYdMV[illegible]/CONSUMER
      <CONSUMER-SECRET>[illegible]2IB4RF
      <access-token>15[illegible]p53HqvYX
      <access-token-secret>[illegible]4ggSzppvTsrLn
   </twitter>
<github>
   <Client-Id>a47e[illegible]94495</Client-Id>
   <Client-Secret>[illegible]5d4fbfb97a5f8f268d2c
</github>
</config>
```

BaseX with Node.js + events

An XQuery chatbot in 50 lines...

1. hello
2. user3: hi
3. user3: xquery 1 to 10
4. basex: 1 2 3 4 5 6 7 8 9 10
5. basex: an event at 2013-02-04T15:22:49.128Z

1. user2: hello
2. hi
3. xquery 1 to 10
4. basex: 1 2 3 4 5 6 7 8 9 10
5. basex: an event at 2013-02-04T15:22:49.128Z

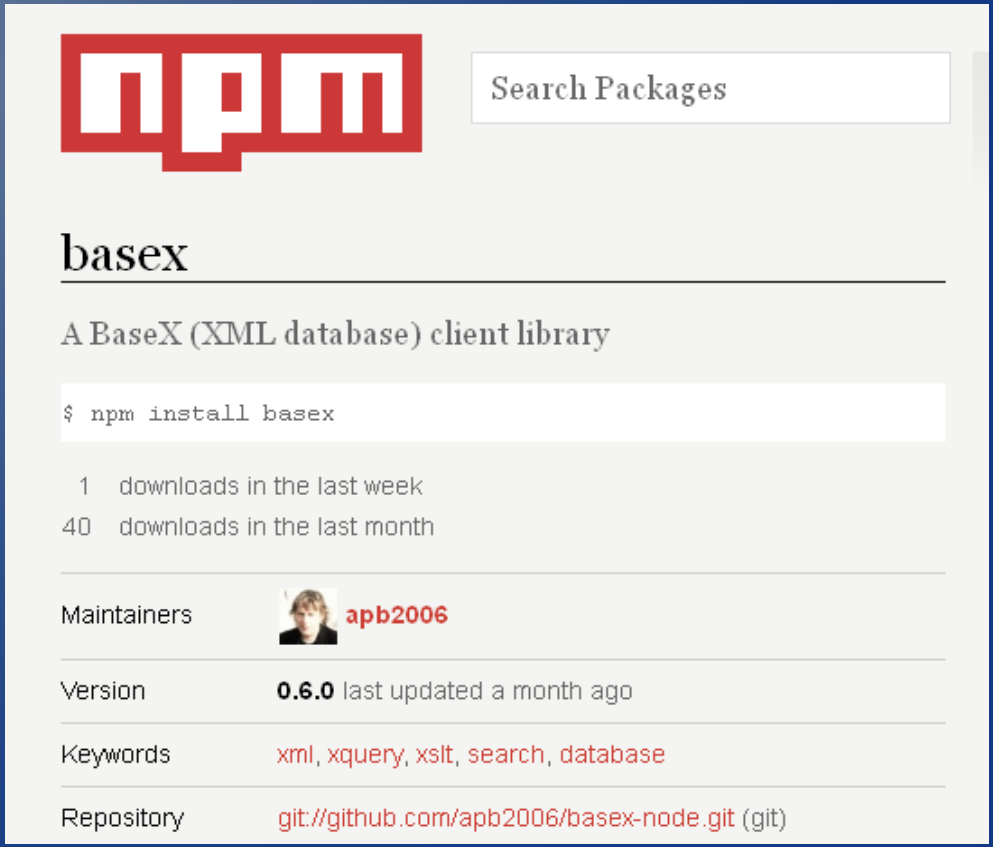
```
C:\Program Files\BaseX\bin>basexclient
Username: admin
Password:
BaseX 7.5.1 beta [Client]
Try help to get more information.

>  xquery db:event("chat","an event at " || current-dateTime())

Query executed in 0.42 ms.
>
```

node.js http://nodejs.org/

- is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications.
- uses an event-driven, non-blocking I/O model that makes it lightweight and efficient
- perfect for data-intensive real-time applications that run across distributed devices.

Basex-node https://github.com/apb2006/basex-node

- is a BaseX client for Node.js
- Uses the BaseX API over a socket
- Available on npm



Basex-node sample

```
1  /*
2   * This example shows how database commands can be executed.
3   */
4  var basex  = require("basex");
5  var client = new basex.Session();
6
7  function callback(err, reply) {
8          var t2=new Date();
9          console.log("Execution completed in ",t2-t0," milliseconds.");
10         console.log(reply.result);
11 };
12 var t0=new Date();
13 client.execute("xquery 1 to 10",callback);
14 client.close(function(){
15     var t2=new Date();
16     console.log("Closed in ",t2-t0," milliseconds.");
17 });
18 var t1=new Date();
19 // not a true time because basex commands not yet done.
20 console.log("Commands send in ",t1-t0," milliseconds.");
21
```

```
C:\temp\chat>node Example.js
Commands send in  0  milliseconds.
Execution completed in  7  milliseconds.
1 2 3 4 5 6 7 8 9 10
Closed in  9  milliseconds.
```

Socket.io http://socket.io/

- Socket.IO is a JavaScript library for realtime web applications.
- a server-side library for node.js
- a client-side library that runs in the browser
- uses the WebSocket protocol, but can fall-back on other methods, such as Adobe Flash sockets, JSONP polling, and AJAX long polling

Chat server based on http://book.mixu.net/ch13.html

```
// Andy bunce jan 2013 based on http://book.mixu.net/ch13.html
var fs = require('fs'),
    http = require('http'),
    sio = require('socket.io');
var port=8001;
var server = http.createServer(function(req, res) {
  res.writeHead(200, { 'Content-type': 'text/html'});
  res.end(fs.readFileSync('./index.html'));
});
server.listen(port, function() {
  console.log('Server listening at http://localhost:',port);
});
// Attach the socket.io server
io = sio.listen(server);
// store messages
var messages = [];
var userId=0;
// Define a message handler
io.sockets.on('connection', function (socket) {
  socket.username="user"+userId++;
  socket.on('message', function (msg) {
    var lmsg=socket.username+ ": "+msg;
    messages.push(lmsg);
    socket.broadcast.emit('message', lmsg);
  });
  // send messages to new clients
  messages.forEach(function(msg) {
    socket.send(msg);
  })
});
```

db:event

http://docs.basex.org/wiki/Database_Module#db:event

- Executes a $query and sends the resulting value to all clients watching the Event with the specified $name. The query may also perform updates; no event will be sent to the client that fired the event.
- db:event($name as xs:string, $query as item()) as empty-sequence()

Add BaseX session

```
var basex=require('basex');
var bx = new basex.Session(ipaddress, basexport, "admin", "admin");
bx.execute("create event chat", afterCreate);
// watch for it
function afterCreate(err, reply) {
  console.log("running afterCreate...");
    if (err)
        console.log("Error: " + err);
    bx.watch("chat", watchCallback);
};
// echo events as chat
function watchCallback(name,msg){
 io.sockets.emit('message', "basex: " +msg);
};
```

Echo Events received

```
    //---- if starts "xquery" execute on basex server -----------
    if(msg.indexOf("xquery ") == 0){
    bx.execute(msg,function(err,reply){
        io.sockets.emit('message', "basex: " +reply.result);
        })
    }
```

Pass messages Starting “xquery “ for execution

Code @ https://gist.github.com/4707605

the End

- http://open1-apb.rhcloud.com/restxq/graphxq
- http://open1-apb.rhcloud.com/cellar
- http://node2-apb.rhcloud.com/