existsolutions

Wednesday 26th October 2011

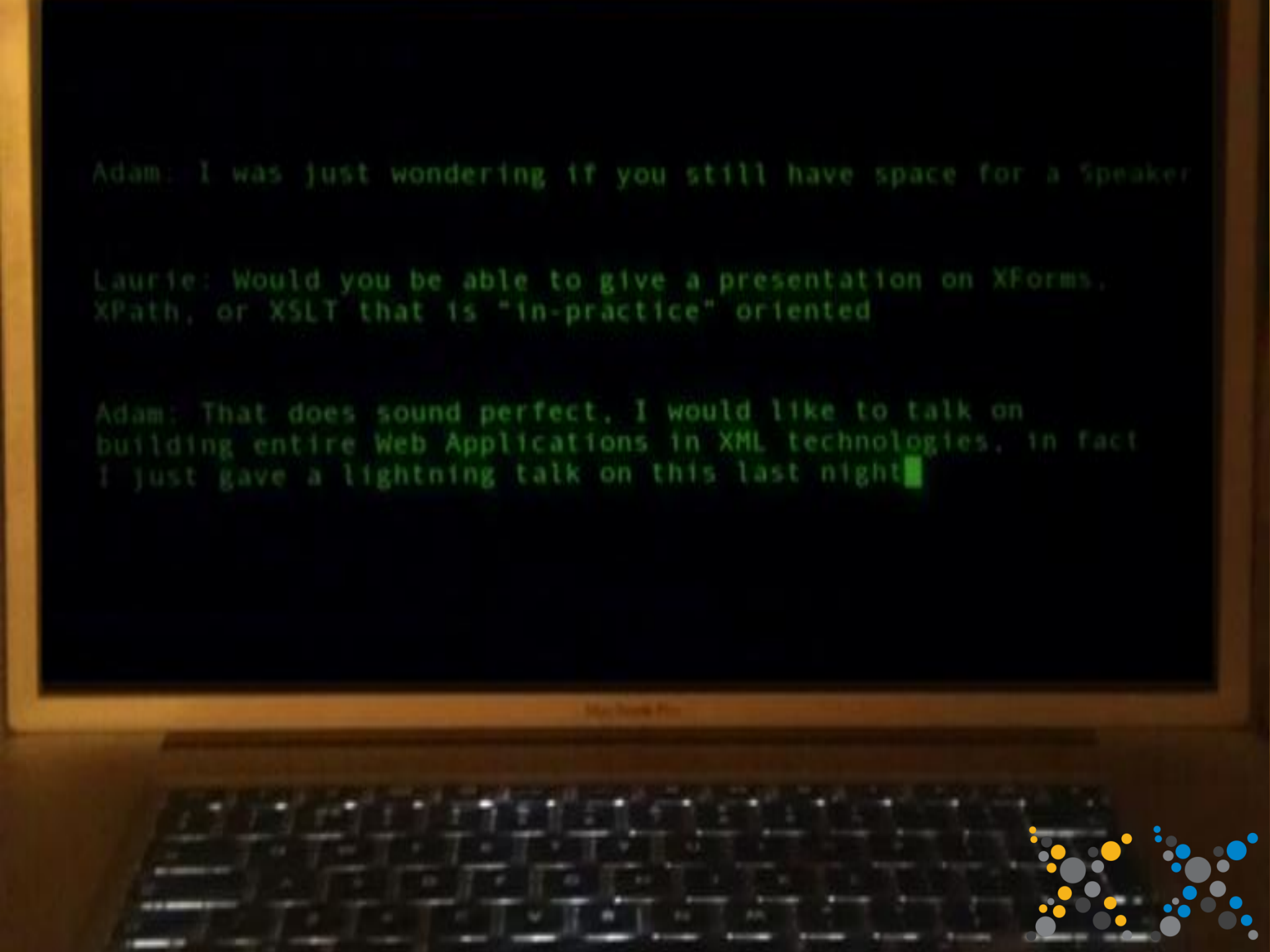
Web Applications and XML Technologies

Adam Retter
adam@existsolutions.com
#adamretter

existdb
www.exist-db.org

Flickr: CC-BY: Hall 1 from Stage by christopher.durant

So, er... how did ‘I’ get here?


Adam: I was just wondering if you still have space for a Speaker
Laurie: Would you be able to give a presentation on XForms, XPath, or XSLT that is "in-practice" oriented
Adam: That does sound perfect, I would like to talk on building entire Web Applications in XML technologies, in fact I just gave a lightning talk on this last night

I'll be right there...

Adam Retter

Graduate Software Engineer

Programming for ~19 years

Building Web Applications for ~13 years

Career:

Not-for-Profit Organisation

Local Government

Private Sector

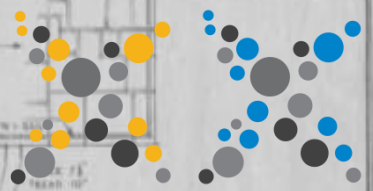
Consultant

Director of eXist Solutions

XML adventures
and
eXist-db contributor

The Plan...

1. Examine the current approach of Web App development
2. Looking at building Web Apps in XML Technologies
3. Conclusions

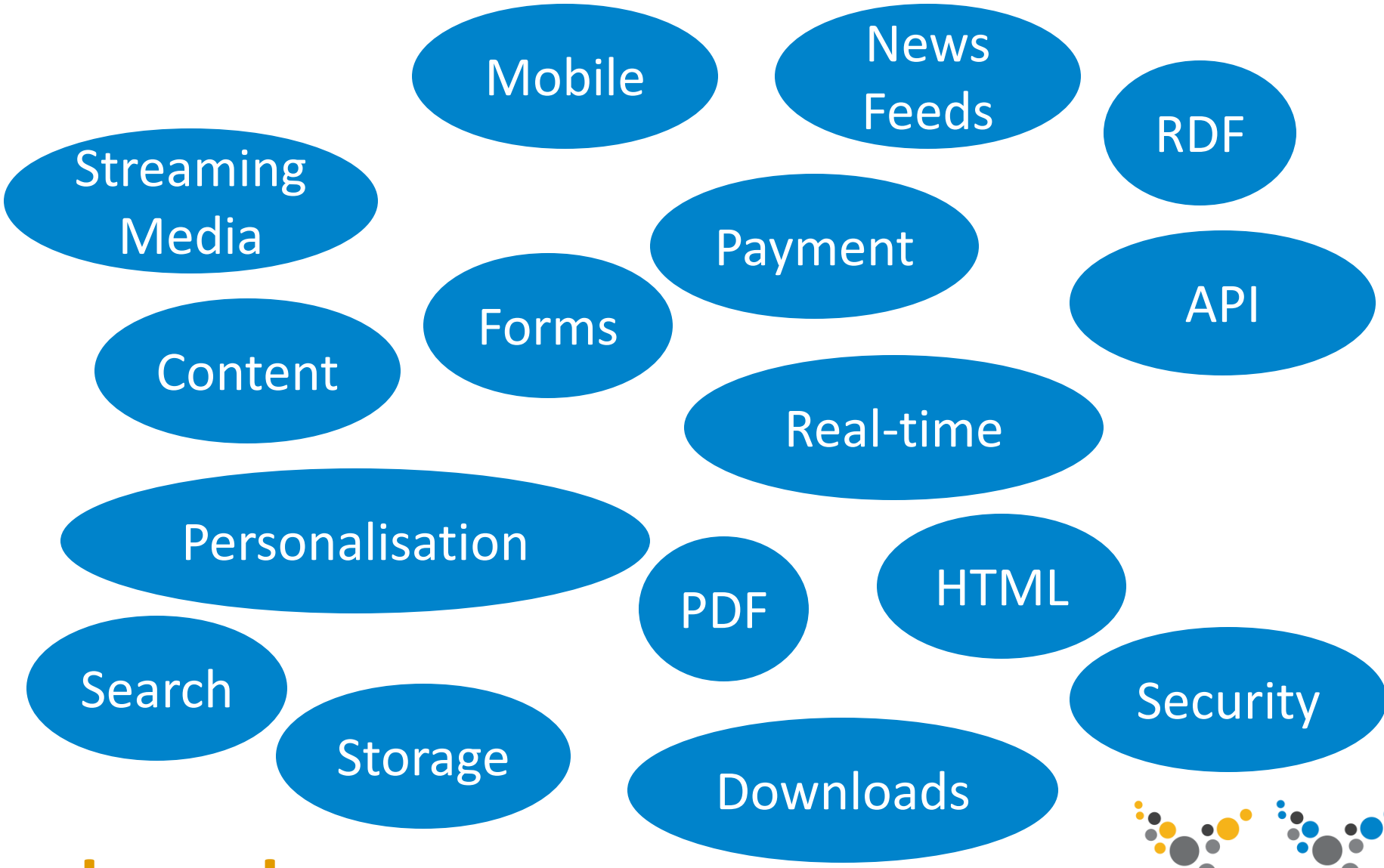


Act 1. Examining the current approach of Web App Development

– How are Web Applications built?

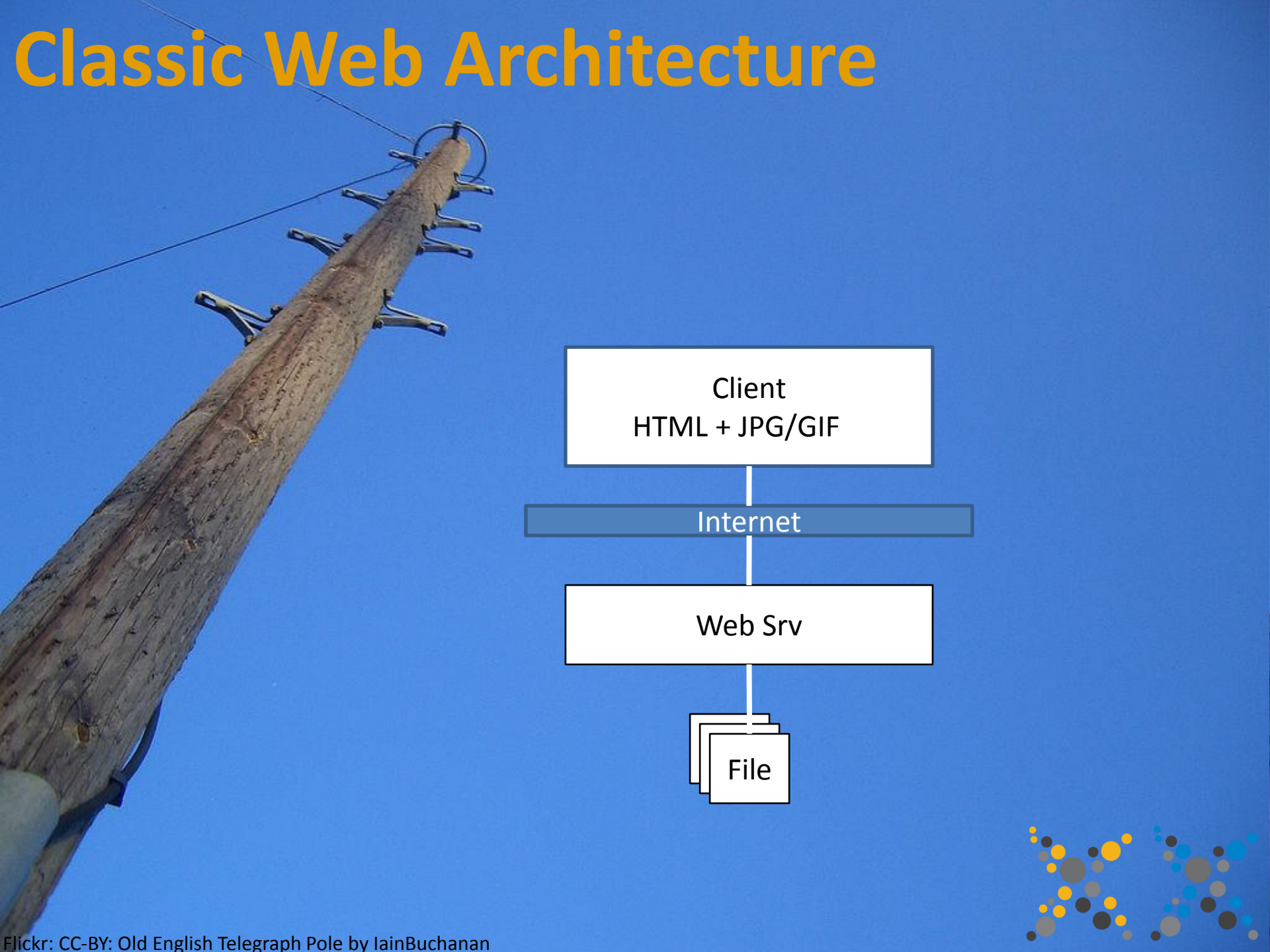
– How have they evolved?

Websites consist of:

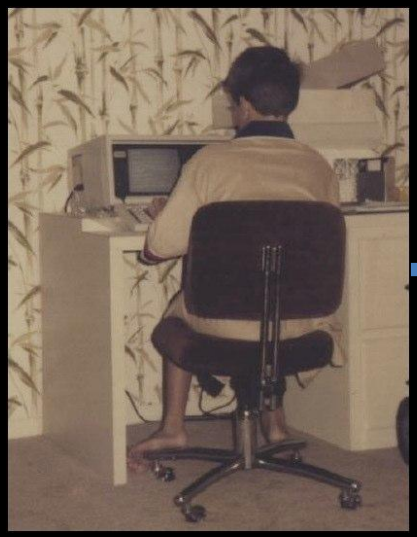


and much more...

Classic Web Architecture



Then the Web took off...



Flickr: CC-BY: The Hacker by s_w_ellis

Web Master

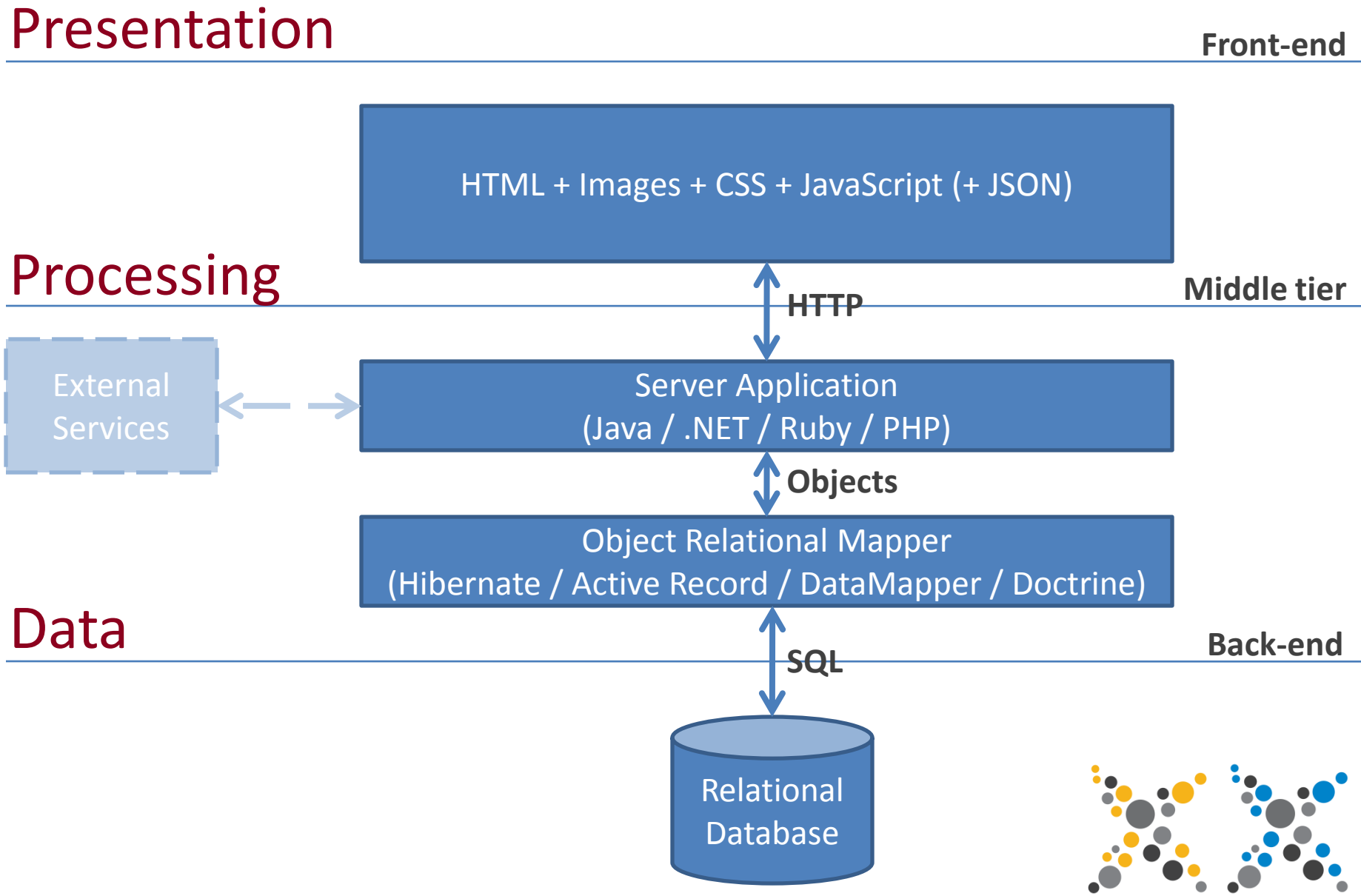
Just one guy?!?

Flickr: CC-BY: New Office by Phillie Casablanca

Web Team

- Content Writers
- Marketing
- Graphic Designers
- Front-end coders
- Back-end coders
- DBAs
- Sys. Admins
- Testers

Modern Web App Architecture

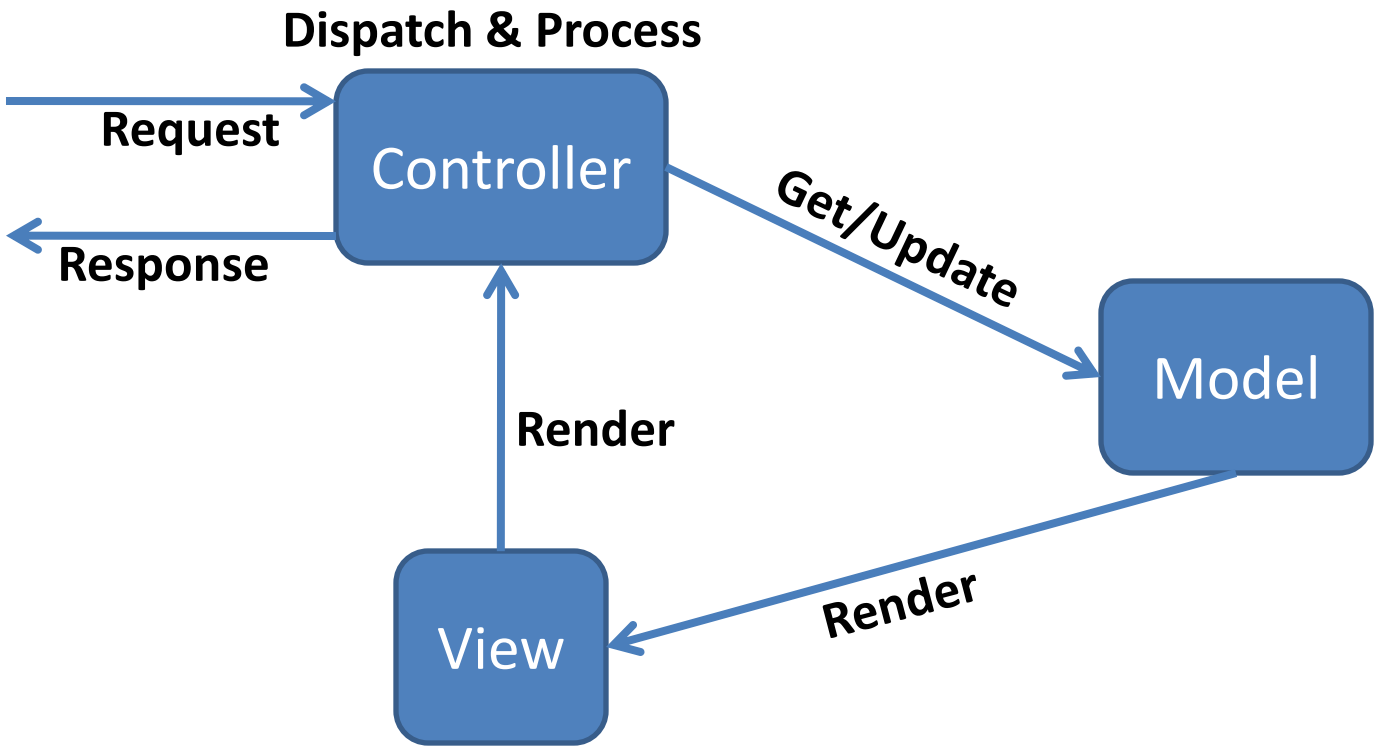


Modern Web App Architecture
Middle-Tier Good Practice

- Decompose into Presentation, Processing and Data
  - Separate business logic, formatting and data
  - MVC / MVP / MVVC approaches and frameworks help
- Frameworks to speed up development time
  - Spring, Facelets, Rails, Symphony, etc. etc.
- Do not embed SQL code in your business code
  - In fact, do not talk to the RDBMS... Use an ORM!

Model View Controller

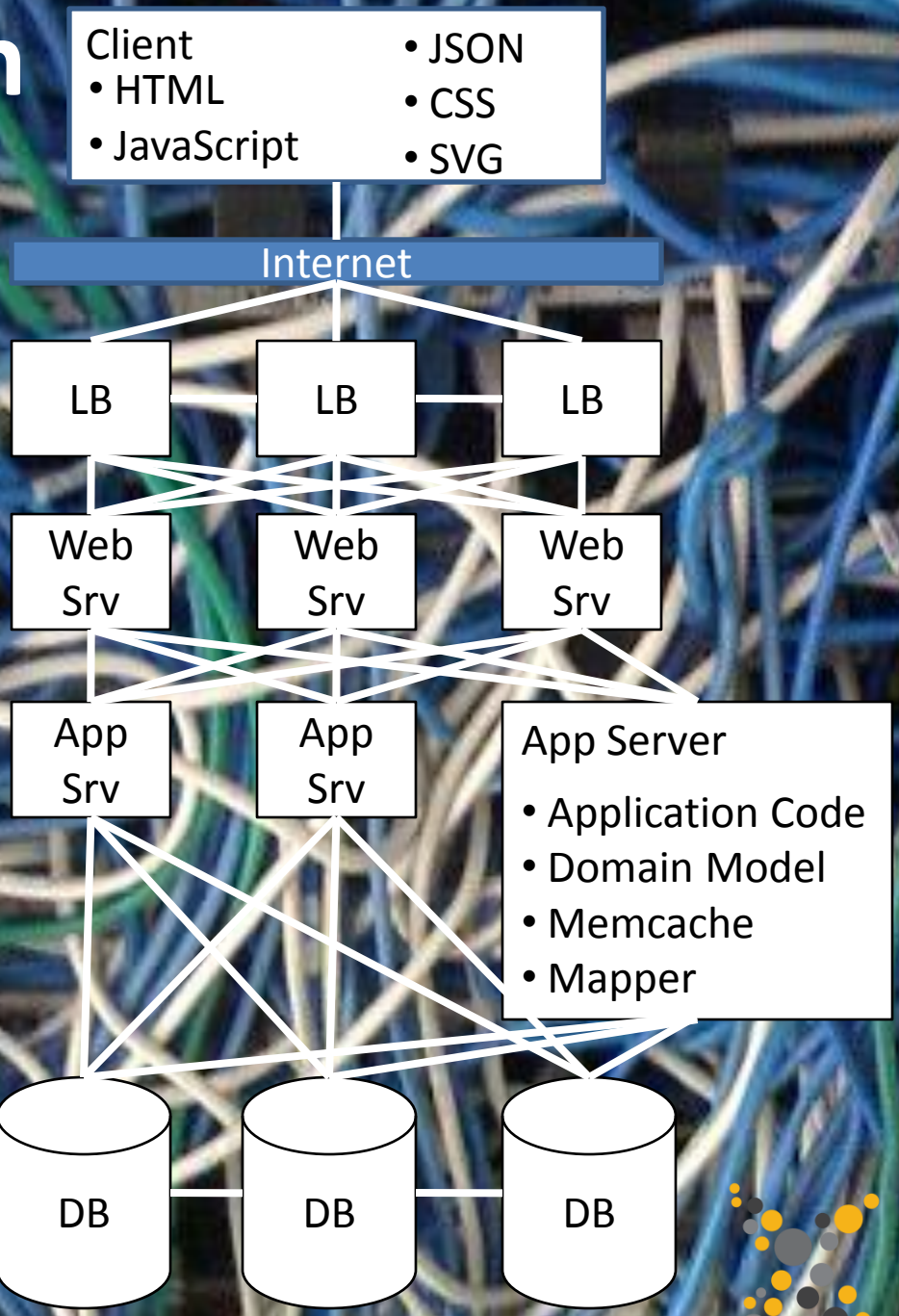
- Its New! Its Old! 1979, Trygve Reenskaug, Xerox PARC
- Its a design pattern, suitable for anything with a UI*
- Separation of Concerns



Modern Web Apps

tear down the wall between client/server

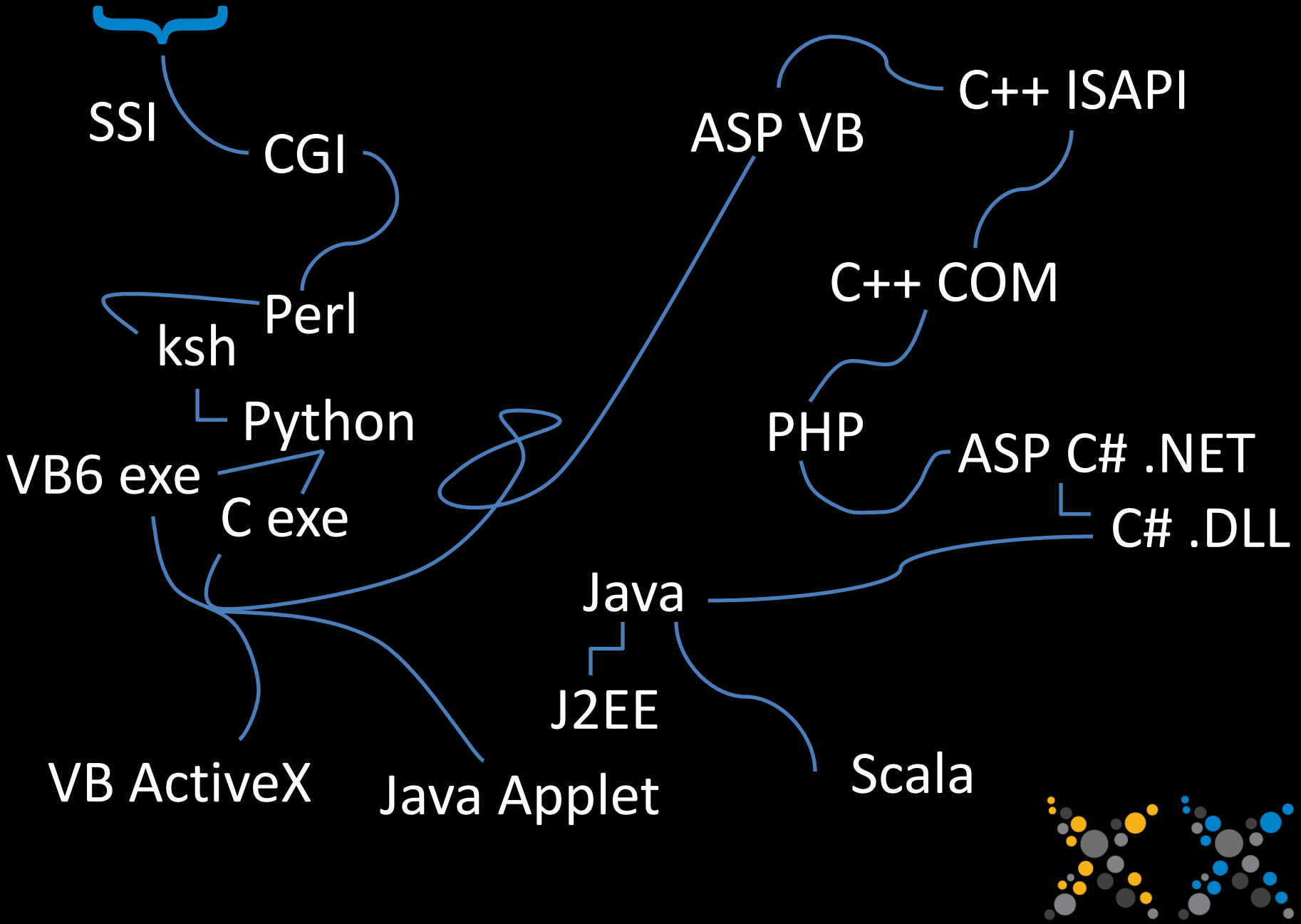
Web Application System Architecture became more complex



Act 2. Looking at building Web Apps in XML Technologies

- Why Act. 1, why not XML?
- XML Web App Technologies
  - XPath
  - XQuery and XSLT
  - XForms

In my past (maybe yours), I built Web apps using...



The Question:

XML is arguably the best approach for exchanging information between systems, considering the Web as the biggest distributed system...

Why is the Web not built on XML?

An Answer – Bad Timing?

SGML

1999	HTML Tags	
1993	HTML Draft	
1995	HTML 2.0 <form/>	
1996	JavaScript 1.0	**XML 1.0** (D)
1997	HTML 3.2 <script/> HTML 4.0 JavaScript 1.2	
1998		XML 1.0 (R) **XSLT 1.0** (D) **XHTML 1.0** (D)
1999	HTML 4.01	**XPath 1.0** XSLT 1.0 (R) XHTML 1.1 (D)
2000		**XForms 1.0** (D)
2001		**XQuery 1.0** (D) XPath 2.0 (D)
2002		XML 1.1 (D)
2003		XForms 1.0 (R)
2004		XML 1.1 (R) XHTML 1.0 (R)
2007	HTML 5 (D)	XQuery 1.0 (R) XPath 2.0 (R) XQuery 3.0 (D)
2009		XForms 1.1 (R)

Maybe you shouldn't worry?

HTML is *almost* XML

...it's close enough that we can '*tidy*' it into XML. (not XHTML)

An Answer – Well, it is a little bit!

XML seen as good for:

- (Web) App Integration (APIs)
  - XML inside SOAP or REST
- (Web) Apps invoke Business Services
  - Service Bus and Messaging (XML)
- Complex (Web) App Configuration
  - Typically a bunch of XML files

Web Applications

We can build entire Web Applications with XML technologies!

...actually, since 2004!

Why build Web Apps in XML technologies?

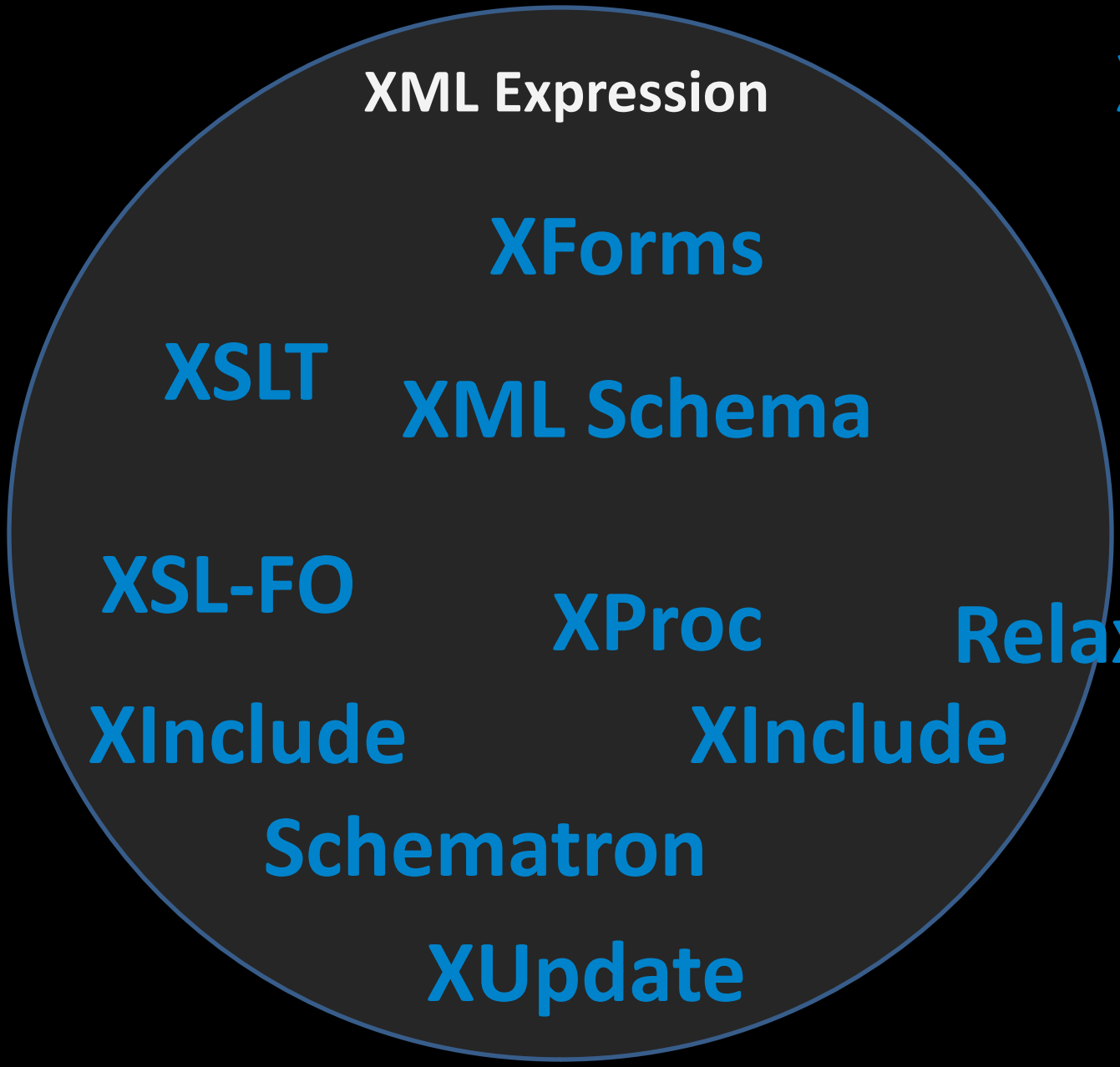
- **(Data) Shape**
  - Already XML or even HTML?
  - Based around documents?
  - Natural tree structure?
  - Hard to model in an RDBMS?
- **Relaxed Schema or Schema Free**
  - Shape of your data is unknown?
  - Schema is flexible or does not exist?
  - Shape is always evolving?
- **Delivery**
  - Main output will be HTML (XHTML) or XML
  - Many different transformations
    (RDF, CSV, JSON, PDF, SVG etc)

How to build Web Apps in XML technologies?

Just like any Web App, adhere to good design:

- Think hard
- Design the Model first
- Plan the Business Services
- Code (Separation of Concerns e.g. MVC)
- Document everywhere
- Test, Test, Test. (TDD is still possible)
- Keep the Model clean
- ...and repeat!

Which XML technologies?



XPath

XPointer

XQuery

DTD

Relax-NG

Which XML technologies to focus on?

XPath (XML Path Language)

- Underpins or used by many other XML specifications

- **XQuery** (XML Query Language)

- **XSLT** (Extensible Stylesheet Language for Transformations)

- **XForms** (“the successor to HTML forms”)

and now for a basic overview...

XPath - XML Path Language

W3C Specification

1.0 recommended in 1999

2.0 recommended in 2009

"XPath gets its name from its use of a path notation as in URLs for navigating through the hierarchical structure of an XML document"

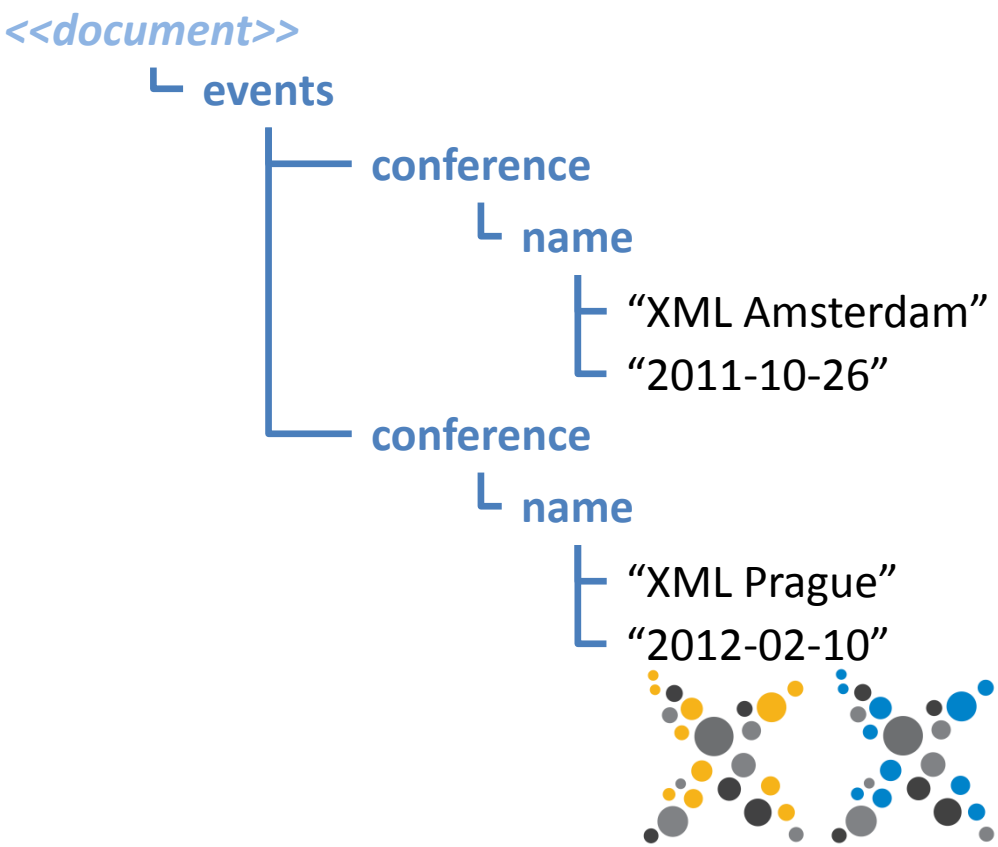
"The primary purpose of XPath is to address parts of an XML document"

Quotes taken from: Chapter 1. Introduction of XML Path Language (XPath) Version 1.0 W3C Recommendation 16 November 1999

XPath Fundamentals

- Consider that an XML Document is really a Tree
  - Describe *how* to reach a Node in the Tree
- Similar to addressing Directories and Files in your Filesystem

```
<?xml version="1.0" encoding="UTF-8"?>
<events>
  <conference ref="xmlams11">
    <name>XML Amsterdam</name>
    <date>2011-10-26</date>
  </conference>
  <conference ref="xmlprg12">
    <name>XML Prague</name>
    <date>2012-02-10</date>
  </conference>
</events>
```



XPath – Path Expressions

- XPath to address the conference(s)?

  /events/conference

Result:

```
<conference ref="xmlams11">
    <name>XML Amsterdam</name>
    <date>2011-10-26</date>
</conference>
<conference ref="xmlprg12">
    <name>XML Prague</name>
    <date>2012-02-10</date>
</conference>
```

- XPath to address the conference(s) name?

  /events/conference/name

Result:

```
<name>XML Amsterdam</name>
<name>XML Prague</name>
```

XPath – Axes

- Allow more complex navigation in the path
  - From the current context (Node)
  - Navigate both Up and Down the Tree

Forward (i.e. down or after)		Backward (i.e. up or before)	
Un-Abbreviated	**Abbreviated**	**Un-Abbreviated**	**Abbreviated**
child	/	parent	..
descendant	.//	ancestor	
following-sibling		preceding-sibling	
following		preceding	
self	.	ancestor-or-self	
descendant-or-self	//		
attribute	@		
namespace			

combine with node or name tests...

XPath – Node Kind and Name Tests

- Allow greater precision in the path expression
  - **Node Kinds**
    - document-node()
    - node()
    - element()
    - attribute()
    - text()
    - comment()
  - **Name Tests**
    - document-node(element(*events*))
    - element(*conference*)
    - attribute(*ref*)

XPath – Axis and Tests Examples

- XPath to address all the name elements

  /descendant-of-self::element(*name*)

  or

  //element(*name*)

  or

  //name

- XPath to get all of the text about all of the conferences

  /child::element(*events*)/child::element(*conference*)
  /descendant-or-self::text()

  or

  /element(*events*)/element(conference)//text()

  or maybe?

  //conference//text()

XPath – Predicates

- Allow you to address an item in a sequence
  - XPath to address the first conference?

    /events/conference[1]

    Result:

```
<conference>
    <name>XML Amsterdam</name>
    <date>2011-10-26</date>
</conference>
```

  - XPath to address the last conference?

    /events/conference[last()]

    or **/events/conference[count(../conference)]**

    Result:

```
<conference>
    <name>XML Amsterdam</name>
    <date>2011-10-26</date>
</conference>
```

XPath – Predicates

Allow you to filter nodes

- XPath to address the name of all conference(s) after 2011?

  /events/conference[date gt “2011-01-01”]/name

  Result:

  ```
  <name>XML Amsterdam</name>
  <name>XML Prague</name>
  ```

- XPath to get the date the conference ‘xmlams11’?

  /events/conference[@id eq “xmlams11”]/date

  Result:

  ```
  <date>2011-10-26</date>
  ```

- Predicates can make use of axes, node and name tests and also *functions and operators.*

XPath – Conclusion

- Simple yet powerful
  - Describe how to access data
  - Not a single line of code for DB access or type conversion!

- XPath 2.0 adds even more
  - sequences
  - if/then/else
  - for loops
  - conditionals: some/every satisfies
  - Separate spec. for many functions and operators
    - W3C XQuery 1.0 and XPath 2.0 Functions and Operators

- For Learning: W3Schools tutorial and W3C specifications

XQuery and XSLT are like twins

XQuery – XML Query Language

W3C Specification
1.0 recommended in 2007
3.0 recommended in 2012?

In almost any situation you could use either!

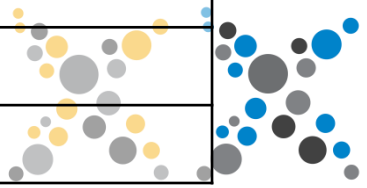
XSLT – XSL for Transformations

W3C Specification
1.0 recommended in 1999
2.0 recommended in 2007
3.0 recommended in 2012?

Why are XQuery and XSLT like twins?

- Similarities:
  - W3C standards with good tools available
  - Built on top of XPath. Data Model, Type System and functions
  - C*an* be used to query XML
  - Transform XML into different outputs
  - Turing complete!
- Differences:

XQuery	**XSLT**
Bespoke Syntax (compact)	XML Syntax (verbose)
Functional Programming	Template Processing
Functions and modules of functions	Imports and overrides
Extensions for Update, Scripting	No Update
Suited to business processing	**Suited to presentation**



XHTML and XQuery vs. XSLT

In XQuery:

```
<p>{current-dateTime()}</p>
```

In XSLT:

```
<p><xsl:value-of select="current-dateTime()"/></p>
```

XQuery – Key Points for Beginners

- Fully functional programming language
- Based around FLWOR statements

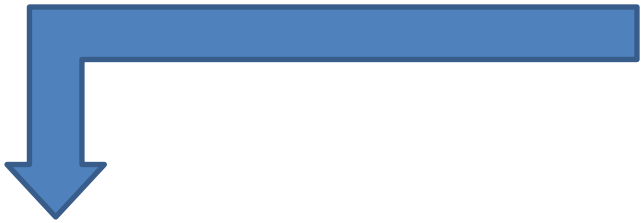
e.g.
```
for $x in doc("mydoc.xml")//conference
let $date := $x/date
where $date lt current-date()
order by $date descending
return $x
```

- Seamless XML processing: Type model is XML type model.
- Easier to learn than XSLT, even though its not XML!
- **Best suited to processing tasks**, use with XSLT for formatting
- Often XQuery impl. provide extensions: XSLT*, XSL-FO, email, FTP, SQL, HTTP, NXDB etc. Also EXPath!
- **Well suited for use with XML Databases**

XQuery example

```
(: Derive the settlement from the conference name and make it explicit :)
declare function local:add-settlement($nodes as node()*) {
   for $node in $nodes return
      typeswitch($node)
         case element(name) return
            (
               $node,
               <settlement>{substring-after($node, " ")}</settlement>
            )
         case element() return
            element {node-name($node)} {
               $node/@*,
               local:add-settlement($node/node())
            }
         default return
            $node
};
local:add-settlement(/events)
```

XQuery example - results



```
<?xml version="1.0" encoding="UTF-8"?>
<events>
   <conference ref="xmlams11">
      <name>XML Amsterdam</name>
      <settlement>Amsterdam</settlement>
      <date>2011-10-26</date>
   </conference>
   <conference ref="xmlprg12">
      <name>XML Prague</name>
      <settlement>Prague</settlement>
      <date>2012-02-10</date>
   </conference>
</events>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<events>
   <conference ref="xmlams11">
      <name>XML Amsterdam</name>
      <date>2011-10-26</date>
   </conference>
   <conference ref="xmlprg12">
      <name>XML Prague</name>
      <date>2012-02-10</date>
   </conference>
</events>
```

XQuery – Tips!

- **Its a Functional Programming Language**
  - Change the way you think about a Program!
  - Think of a function as transforming input to output
  - You cant have global/shared state – don't fight it!
    - Recursion and Function Passing are the answer
  - Operate on Sequences where possible, do not loop!
    - Better optimisations possible
- **Know your Processor**
- **Just because you can, doesn't mean...**

Flickr: CC-BY: Transformer by Wakko

XSLT – Key Points for Beginners

- Transformations part of the Extensible Stylesheet Languages
- Based around Templates, define your own templates and functions.

e.g.

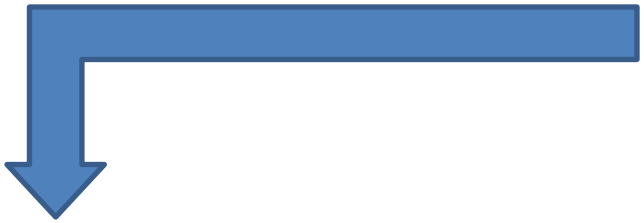
```
<xsl:stylesheet version="2.0“>
    <xsl:template match="events">
        <xsl:apply-templates/>
    </xsl:template>
    <xsl:template match="conference[date lt current-dateT
        <xsl:value-of select="name"/>
    </xsl:template>
</xsl:stylesheet>
```

- Seamless XML processing: type model is XML type model.
- It’s XML!
- **Best suited to** presentation tasks, e.g. generate HTML.
- Often XSLT impl. provide extensions: SQL, HTTP, Also EXPath!
- **Well suited for use in processing pipelines**

XSLT example

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="2.0">
   <!-- Derive the settlement from the conference name and make it explicit-->

   <xsl:template match="events">
      <xsl:apply-templates/>
   </xsl:template>

   <xsl:template match="name">
      <xsl:copy><xsl:apply-templates/></xsl:copy>
      <settlement><xsl:value-of select="substring-after(., ' ')"/></settlement>
   </xsl:template>

   <xsl:template match="node() | @*">
      <xsl:copy>
         <xsl:apply-templates select="node() | @*"/>
      </xsl:copy>
   </xsl:template>

</xsl:stylesheet>
```

XSLT example - results



```
<?xml version="1.0" encoding="UTF-8"?>
<events>
   <conference ref="xmlams11">
      <name>XML Amsterdam</name>
      <settlement>Amsterdam</settlement>
      <date>2011-10-26</date>
   </conference>
   <conference ref="xmlprg12">
      <name>XML Prague</name>
      <settlement>Prague</settlement>
      <date>2012-02-10</date>
   </conference>
</events>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<events>
   <conference ref="xmlams11">
      <name>XML Amsterdam</name>
      <date>2011-10-26</date>
   </conference>
   <conference ref="xmlprg12">
      <name>XML Prague</name>
      <date>2012-02-10</date>
   </conference>
</events>
```

...Exactly the same as the XQuery results :-)

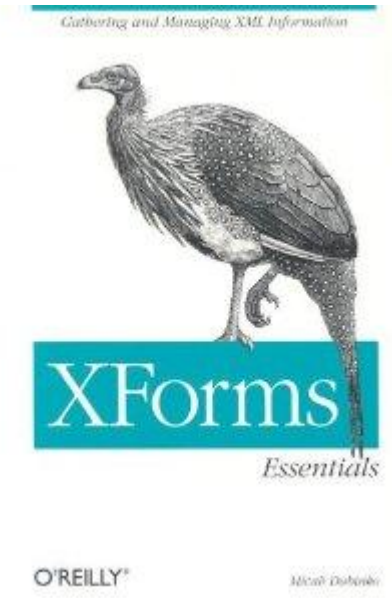
XSLT – Tips!

- (IMHO) **It is not a programming language**
  - Thinks in terms of templates
    - Keep templates small: Single Responsibility Pattern
    - Transforming input to output
    - Don’t loop where you could use template matches
- **Start with an Identity Transform Pattern**
- **Logical layout – Follow Source or Dest. struct.**
- **Know your Processor**
- **Just because you can, doesn’t mean..,**

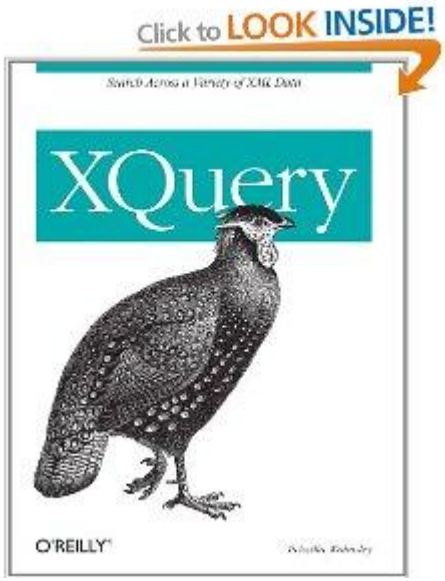
Flickr: CC-BY: 2007 California Corporate Tax, shows penalty by Casey Serin

XForms – Key Points for Beginners

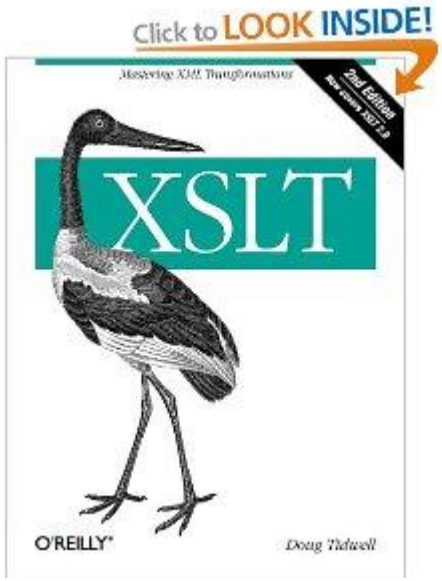
- It's Simple and Fun!
  - This is *proven* by the fact that its the smallest XML book ;-)



240 Pages
1.5 CM thick



512 Pages
2.3 CM thick



988 Pages
4.3 CM thick

...XForms is 3X simpler than XSLT ;-)

XForms – Key Points for Beginners

- It's XML! It's *NOT* called XML Forms!!!
- **Not Standalone**. Embeds in a container
- Similar to HTML Forms *but* you get an **XML document as the result**!
- Enforces **Separation of Concerns**
  - **Data Model** – Declare your XML instance(s) structure
  - **Controls** – Your form input/output boxes/buttons etc
  - **Bindings** – Link the Controls to the Data Model
- Enforce field/data requirements
- Forms can be *complex*!
- **Needs a Processor!**

Not XForms – A typical HTML Form

```
<form method="post" action="http://server.com/conference.php?action=add">
        <fieldset>
           <label for="name">Conference Name:</label>
           <input type="text" id="name"/>

           <label for="date">Conference Date:</label>
           <input type="text" id="date"/>

          <input type="submit" name="save" value="Save"/>
        </fieldset>
      </form>
```

At submit, server receives:

```
POST http://server.com/conference.php?action=add
Content-Type: application/x-www-form-urlencoded

name=XML%20Amsterdam&date=October%2026%202011
```

XForms example

```
<html>
  <head>

    <xf:model>
      <xf:instance>
        <events xmlns="">
          <conference>
            <name/>
            <date/>
          </conference>
        </events>
      </xf:instance>

      <xf:bind nodeset="conference/name" required="true()"/>
      <xf:bind nodeset="conference/date" type="date" required="true()"/>

      <xf:submission id="s-send" method="post" resource="http://server.com/conference?actior
    </xf:model>

  </head>
```

HTML Container

Instance

Bindings

XForms example (continued)

```
<body>
  <xf:group>
    <xf:label>Conference registration form</xf:label>
    <xf:input ref="conference/name" incremental="true">
      <xf:label>Conference name:</xf:label>
      <xf:hint>Enter the name of the Conference</xf:hint>
      <xf:alert>You must enter a name for the Conference</xf:alert>
    </xf:input>
    <xf:input ref="conference/date" incremental="true">
      <xf:label>Conference Date:</xf:label>
      <xf:hint>Enter the start date of the Conference</xf:hint>
      <xf:alert>You must enter a date for the Conference
      in the format "2010-12-25"</xf:alert>
    </xf:input>
    <xf:trigger src="images/icons/btn_save.gif">
      <xf:label>Save</xf:label>
      <xf:hint>Saves the conference details</xf:hint>
      <xf:send submission="s-send"/>
    </xf:trigger>
  </xf:group>
</body>
</html>
```

Controls

XForms example (continued)

At submit, server receives:

```
POST http://server.com/conference?action=add
Content-Type: text/xml

<events>
   <conference ref="xmlams11">
      <name>XML Amsterdam</name>
      <date>2011-10-26</date>
   </conference>
</events>
```

Some XForms features

- Triggers – Do Something, submit / state-change

```
<xf:trigger>
   <xf:send submission="s-send"/>
</xf:trigger>
```

- Repeats – Easily collect something N times

```
<xf:repeat nodeset="conference">
    <xf:input ref="name"/>
    <xf:input ref="date“/>
</xf:repeat>
```

- Switches – Triggers can change the form

```
<xf:switch>
 <xf:case id="in" selected="true">
  <xf:input ref="yourname">
 </xf:case>
 <xf:case id="in" selected=“false">
  <xf:output ref=“yourname">
 </xf:case>
</xf:switch>
```

XForms Tips

- **Start Simple!**
- Do not reinvent the wheel: XML Schema for typing.
- Test troublesome complex forms as small snippets.
- Be plentiful with xf:alert and xf:hint
  - **Use human understandable messages**
- Instance from a URI; Same Form for Create or Edit
- Well Suited to RESTful environments
- **Always use a Server-Side processor**

But, where does all the XML live?

Flickr: CC-BY: Forest Landscape by Van Pelt

But where does the XML live?

IN THE *PAST*... BAD THINGS HAPPENED!

- Lots of XML files, let's store them in the Filesystem
- We have a team of expensive RDBMS DBAs –
  - Store XML in BLOB/CLOB and some metadata
  - Shred XML into RDBMS
    - Manually – complex custom code
    - Automatically – (req. Schema) = big table mess
  - RDBMS introduced XMLType and XMLTable
    - Querying becomes near impossible!

XML lives in a Native XML Database

- The right tool for the right job
- It understands XML!
- Logical model of Collections, Documents and Nodes
- Provides Store/Retrieve/Query/Transform
- Provides XML optimisations
  - Indexing of values and full-text
  - Very fast querying!
- Can often talk to traditional RDBMS

XML and the Web

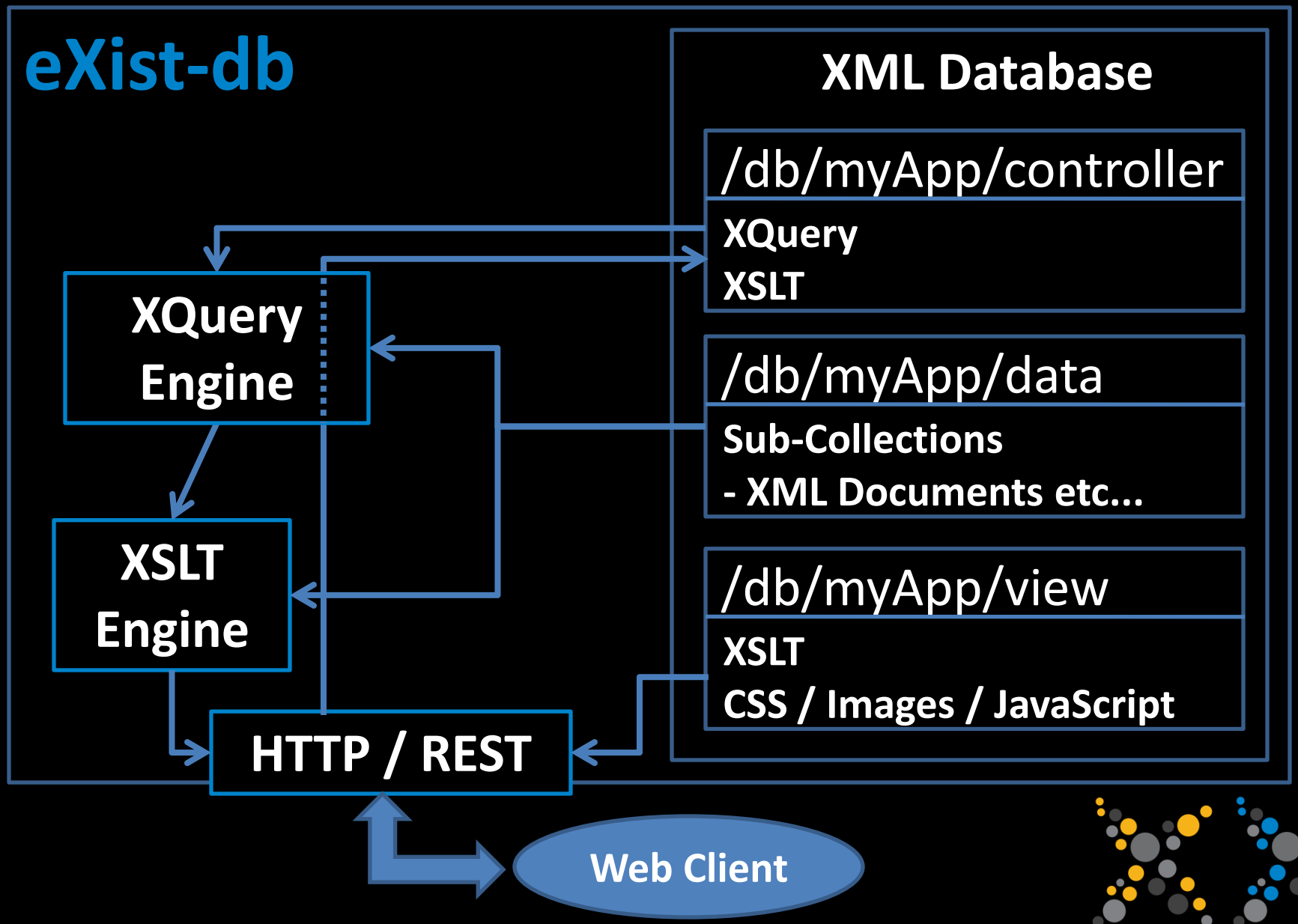
So we have:

- XML for the Data Model
- Native XML Database to store
- XML technologies to process

What about the web?

- Your XML Database / Application Platform delivers this all over HTTP

XML and the Web



What is XRX?

A Web Application Architecture

XForms

REST

XQuery

XRX is not just XRX!

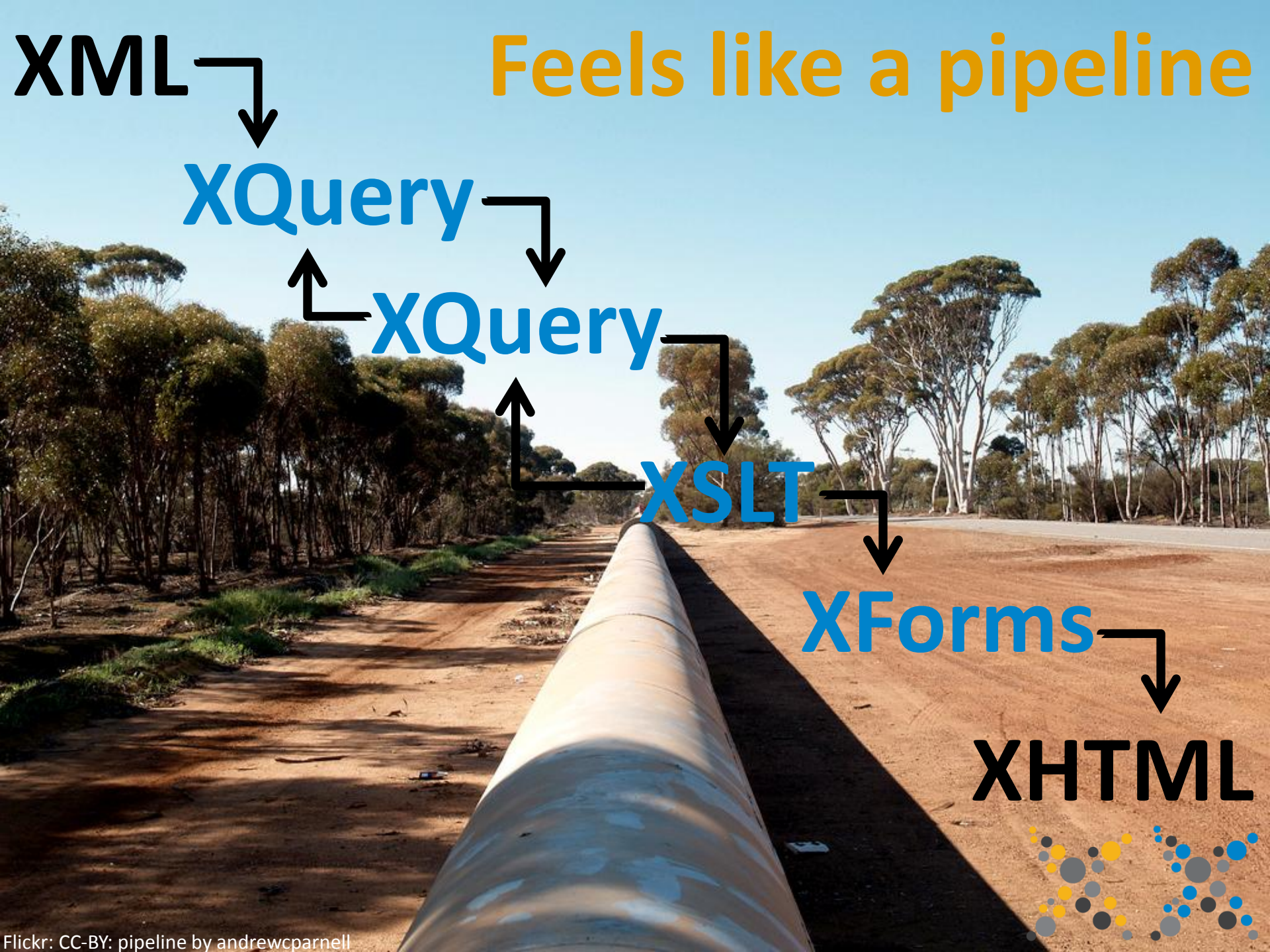
++

XSLT
XInclude
XSL-FO
EXPath
XUpdate
NXDB

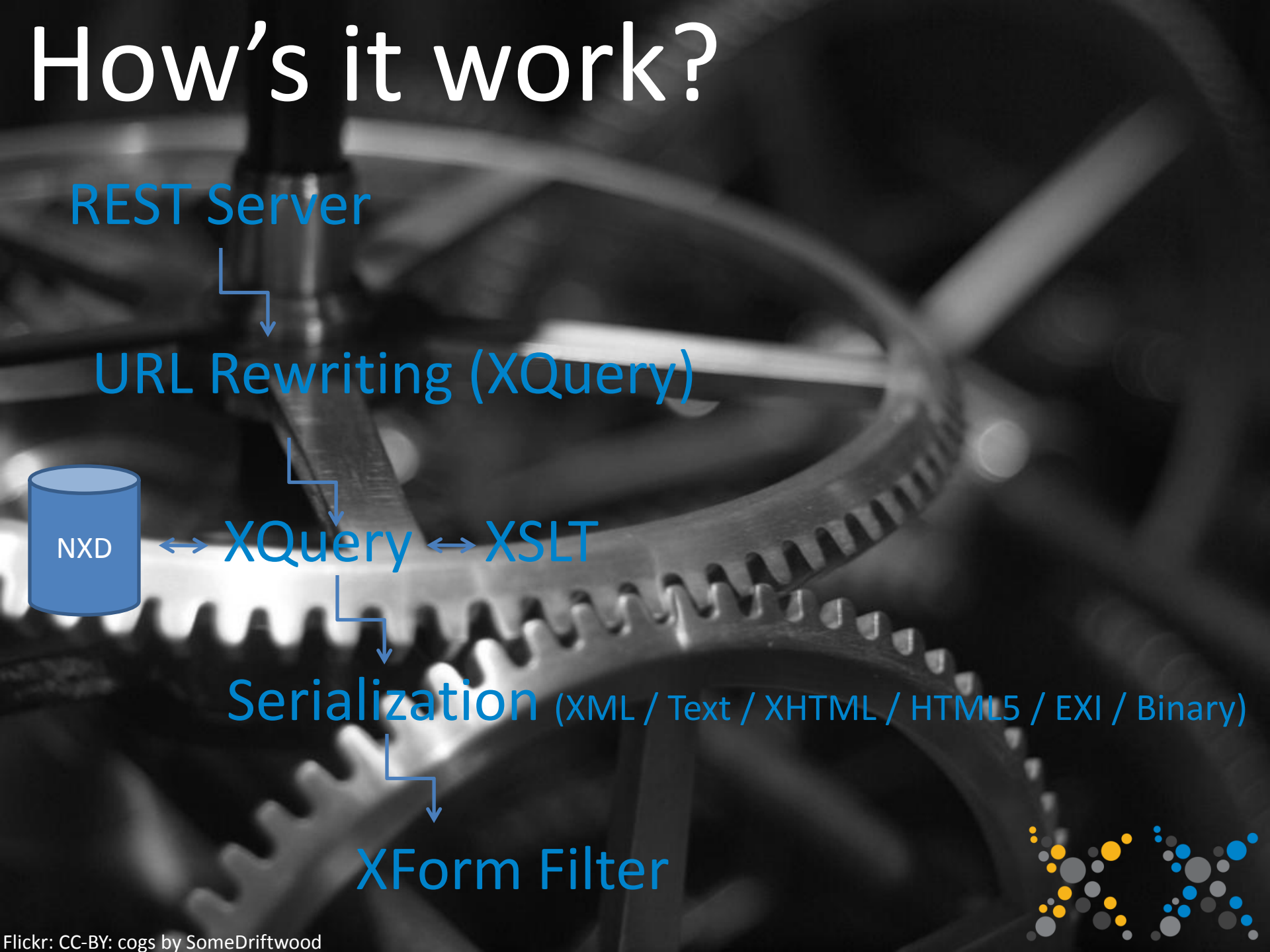
...or any/all XML technologies and REST

The XRX+ house

When everything is XML
its like building with Lego



How's it work?



++ some extensions

Standardise the extensions

EXPath

www.expath.org

Flickr: CC-BY: cogs by SomeDriftwood

But, what about...

- Layering/Separation of Concerns
  - Naturally: XHTML/XForms <- XSLT <- Xquery
    - You do not need data translation!
  - MVC: You can use XQMVC or Mustache or...
- Cool stuff in the browser (jQuery etc)
  - XML to JSON serialization
  - jquery.xqm – jQuery from XML grammar
  - XQiB – XQuery in the browser
    - XQuery everywhere!
    - Bindings to DOM and frameworks (e.g. jQuery)

Act 3. Conclusions

– Why XML for Web Apps?

– Demo

– Q & A

XPath/XQuery/XSLT/XProc/XForms

No barriers between Data and Processing

No barriers between Data and Processing

IMPORTANT:

This is the key to our simplicity!

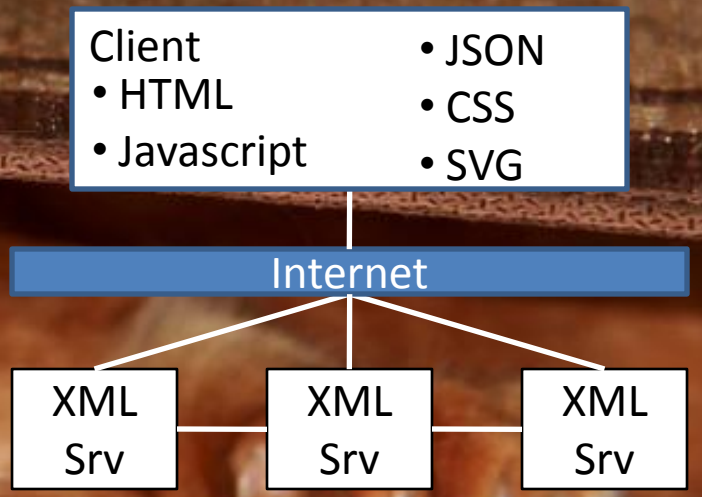
3 become 1, for example:

PHP
Doctrine
SQL

XQuery

Your stack does not have to be complex...

Flatter is better!



Fast development

No Data Access Layer/Mapper

~ 57% LoC Reduction (ETH*)

Picture from Peninsula Humane Society

XQuery Web Apps make you feel Cool

Rapid

Easy

Agile

Safe

But the Kids...

Flickr: CC-BY: Cool Cat by Eleazar Paradise

The Demo

Java, JSF, Spring, Hibernate	XML Technologies
925 Lines of Java	105 Lines of XQuery
652 Lines of XML	70 Lines of XSLT
165 Lines of XHTML	105 Lines of XForms
	15 Lines of XHTML
1742 Lines **1 Hour 24 minutes**	**295 Lines** **> 30 minutes**