XML Programming

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - XML
  - XML programming
  - AJAX revisited

Agenda

- **XML**
- XML programming
- AJAX revisited

XML

- Preliminary observation
  - Much of the world's information is:
    - Textual
    - Hierarchical

XML

- ***XML (Extensible Markup Language)***
  - A language for expressing **textual** documents in **hierarchical** (tree-structured) form
  - Worldwide Web Consortium (W3C) specification
  - Similar to HTML…

XML

- XML vs. HTML
  - ***Empty elements*** must be expressed using self-closing tags
    - Empty element: has start tag and no end tag
    - `<hr>`
      - Well formed in HTML
      - Malformed in XML
    - `<hr />`
      - Well formed in HTML
      - Well formed in XML

XML

- XML vs. HTML (cont.)
  - Element nesting must be proper
    - `<strong>…<em>…</strong>…</em>`
      - Well formed (or at least acceptable) in HTML
      - Malformed in XML
    - `<strong>…<em>…</em>…</strong>`
      - Well formed in HTML
      - Well formed in XML

XML

- XML vs. HTML (cont.)
  - Attribute values must be quoted
    - `<span id=myspan>Something</span>`
      - Well formed in HTML
      - Malformed in XML
    - `<span id="myspan">Something</span>`
      - Well formed in HTML
      - Well formed in XML

XML

- XML vs. HTML (cont.)
  - Allows processing instructions
    - `<? ProcessingInstruction ?>`
    - Provide information to XML processor about how to handle the XML document

XML

- XML vs. HTML (cont.)
  - **You define the tags!**

XML

- See **books.html**
  - Tag set is predefined
- See **books.xml**
  - Tag set is **not** predefined

XML

- Some theory:
  - ***Content***
    - The document's raw data
  - ***Semantic structure***
    - The **organization** of the content
  - ***Presentation***
    - The **rendering** of the content

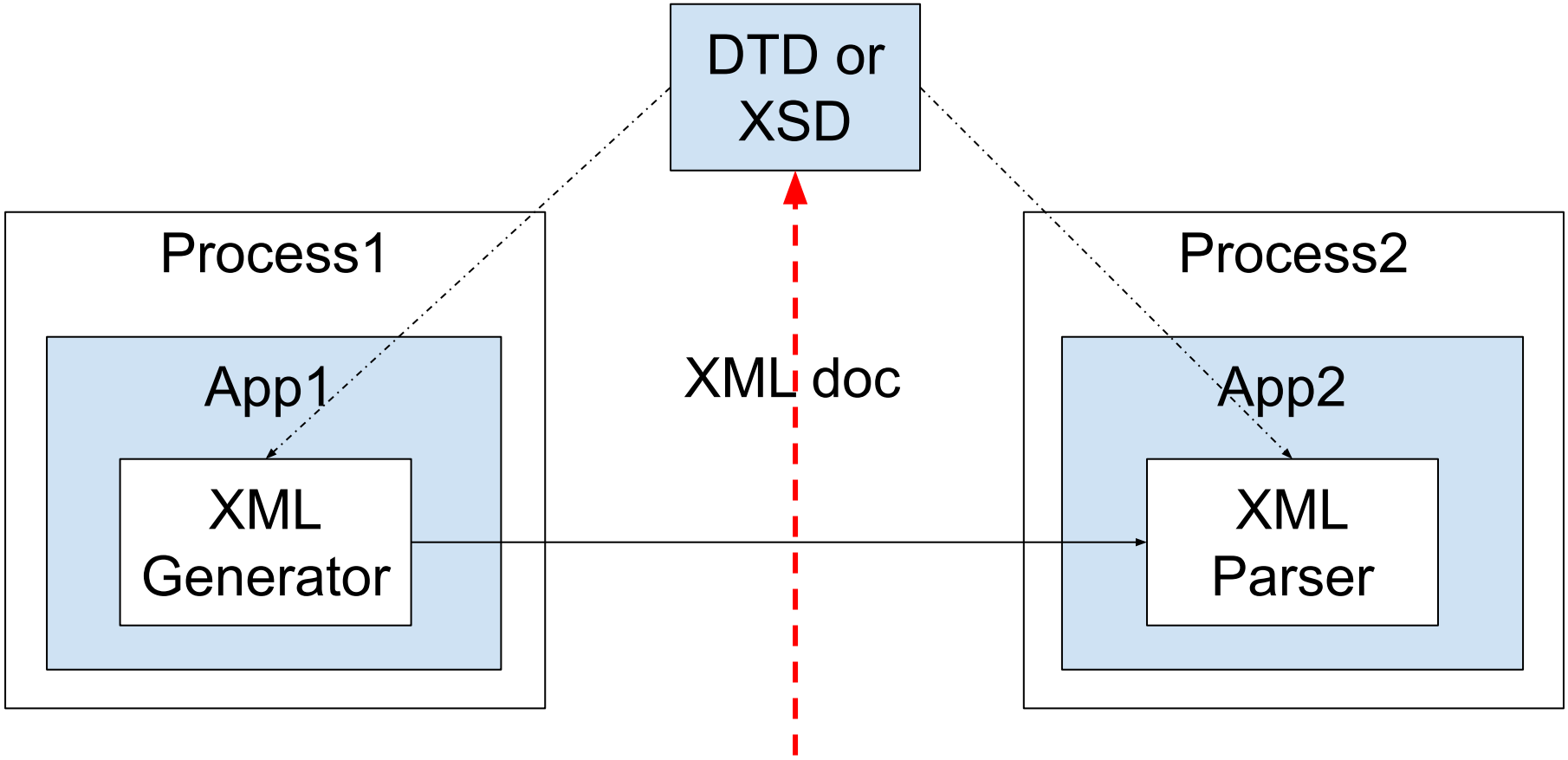
XML

- HTML
  - Tags define **presentation**
  - **Semantic structure** is absent
- XML
  - Tags define **semantic structure**
  - **Presentation** is absent

XML

- Applications of XML:
  - Publishing
    - See **Appendix 1**
  - Data communication
    - Covered now…

XML



Document Type Definition (DTD)
XML Schema Definition (XSD)

Agenda

- XML
- **XML programming**
- AJAX revisited

XML Programming

- Examples in this lecture:
  - In Python
    - Appropriate for XML programming on the **server-side** of a Web app
  - In JavaScript
    - Appropriate for XML programming on the **client-side** of a Web app (i.e., in a browser)

XML Programming

- To run the example **JavaScript** programs in this lecture:
  - `npm install xmldom`
  - `npm install xmlserializer`
  - `npm install sax-parser`

XML Programming

- **World Wide Web Consortium (W3C)** defines two standard APIs:
  - ***SAX***: The **S**imple **A**PI for **X**ML
    - See **Appendix 2**
  - ***DOM***: The **D**ocument **O**bject **M**odel
    - Covered now…

XML Programming

- A DOM parser:
  - Parses given XML doc in its entirety
  - Builds a DOM tree
    - An in-memory tree of objects/nodes

XML Programming

- **writedom** programs
  - Read a XML doc, write its entire DOM

XML Programming

- See **writedom.py**, **domfrompython.txt**

```
$ python writedom.py books.xml > domfrompython.txt
$ cat domfrompython.txt
Document: #document=None
   Comment: #comment=
============================================================
   Comment: #comment= books.xml
   Comment: #comment= Author: Bob Dondero
   Comment: #comment=
============================================================
…
```

XML Programming

- See **writedom.js**, **domfromjavascript.txt**

```
$ node writedom.js books.xml > domfromjavascript.txt
$ cat domfromjavascript.txt
Document: #document=None
   Processing Instruction: undefined=version="1.0"
   Text: #text=WHITESPACE
   Comment: #comment=
============================================================
   Text: #text=WHITESPACE
   Comment: #comment= books.xml
   Text: #text=WHITESPACE
   Comment: #comment= Author: Bob Dondero
   Text: #text=WHITESPACE
   Comment: #comment=
============================================================
…
```

XML Programming

- **writebooks** programs
  - Write all books in books.xml

XML Programming

- See **writebooks.py**

```
$ python writebooks.py
ISBN: 123
Author: Kernighan
Title: The Practice of Programming
Price: 40.74 dollars

ISBN: 234
Author: Kernighan
Title: The C Programming Language
Price: 24.99 dollars

ISBN: 345
Author: Sedgewick
Title: Algorithms in C
Price: 61.59 dollars

$
```

XML Programming

- See **writebooks.js**

```
$ node writebooks.js
ISBN: 123
Author: Kernighan
Title: The Practice of Programming
Price: 40.74 dollars

ISBN: 234
Author: Kernighan
Title: The C Programming Language
Price: 24.99 dollars

ISBN: 345
Author: Sedgewick
Title: Algorithms in C
Price: 61.59 dollars

$
```

XML Programming

- **writebooksshort** programs
  - Same as writebooks programs

XML Programming

- See **writebooksshort.py**

```
$ python writebooksshort.py
ISBN: 123
Author: Kernighan
Title: The Practice of Programming
Price: 40.74 dollars

ISBN: 234
Author: Kernighan
Title: The C Programming Language
Price: 24.99 dollars

ISBN: 345
Author: Sedgewick
Title: Algorithms in C
Price: 61.59 dollars

$
```

XML Programming

- See **writebooksshort.js**

```
$ node writebooksshort.js
ISBN: 123
Author: Kernighan
Title: The Practice of Programming
Price: 40.74 dollars

ISBN: 234
Author: Kernighan
Title: The C Programming Language
Price: 24.99 dollars

ISBN: 345
Author: Sedgewick
Title: Algorithms in C
Price: 61.59 dollars

$
```

XML Programming

- **roundtripxml** programs
  - **Parse**: XML doc → DOM tree
    - Common
  - **Generate**: DOM tree → XML doc
    - Less common

XML Programming

- See **roundtripxml.py**

```
$ python roundtripxml.py books.xml
<?xml version="1.0" ?><!--
================================================================= --><!--
books.xml                                                         --><!--
Author: Bob Dondero                                               --><!--
================================================================= --><books>
   <book>
      <isbn>123</isbn>
      <author>Kernighan</author>
      <title>The Practice of Programming</title>
      <price currency="dollars">40.74</price>
   </book>
   <book>
      <isbn>234</isbn>
      <author>Kernighan</author>
      <title>The C Programming Language</title>
      <price currency="dollars">24.99</price>
   </book>
   <book>
      <isbn>345</isbn>
      <author>Sedgewick</author>
      <title>Algorithms in C</title>
      <price currency="dollars">61.59</price>
   </book>
</books>
$
```

XML Programming

- See **roundtripxml.js**

```
$ node roundtrip.xml.js books.xml

ERROR
```

XML Programming

- See **roundtripxml.js** (cont.)

```
$ node roundtripxml.js books.xml
<!-- ============================================================= -->
<!-- books.xml                                                     -->
<!-- Author: Bob Dondero                                           -->
<!-- ============================================================= -->

<books xmlns="undefined">
   <book>
      <isbn>123</isbn>
      <author>Kernighan</author>
      <title>The Practice of Programming</title>
      <price currency="dollars">40.74</price>
   </book>
   <book>
      <isbn>234</isbn>
      <author>Kernighan</author>
      <title>The C Programming Language</title>
      <price currency="dollars">24.99</price>
   </book>
   <book>
      <isbn>345</isbn>
      <author>Sedgewick</author>
      <title>Algorithms in C</title>
      <price currency="dollars">61.59</price>
   </book>
</books>
$
```

After removing the processing instruction from books.xml

Agenda

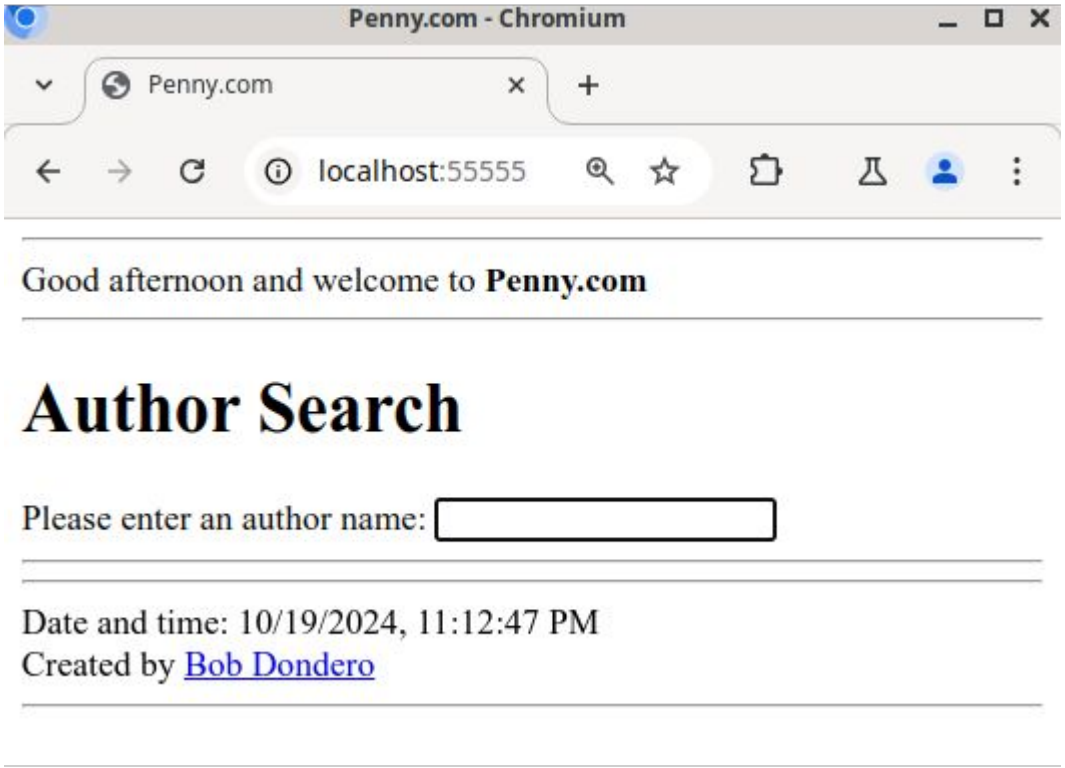
- XML
- XML programming
- **AJAX revisited**

AJAX Revisited

- **AJAX**
  - Asynchronous JavaScript and **XML**

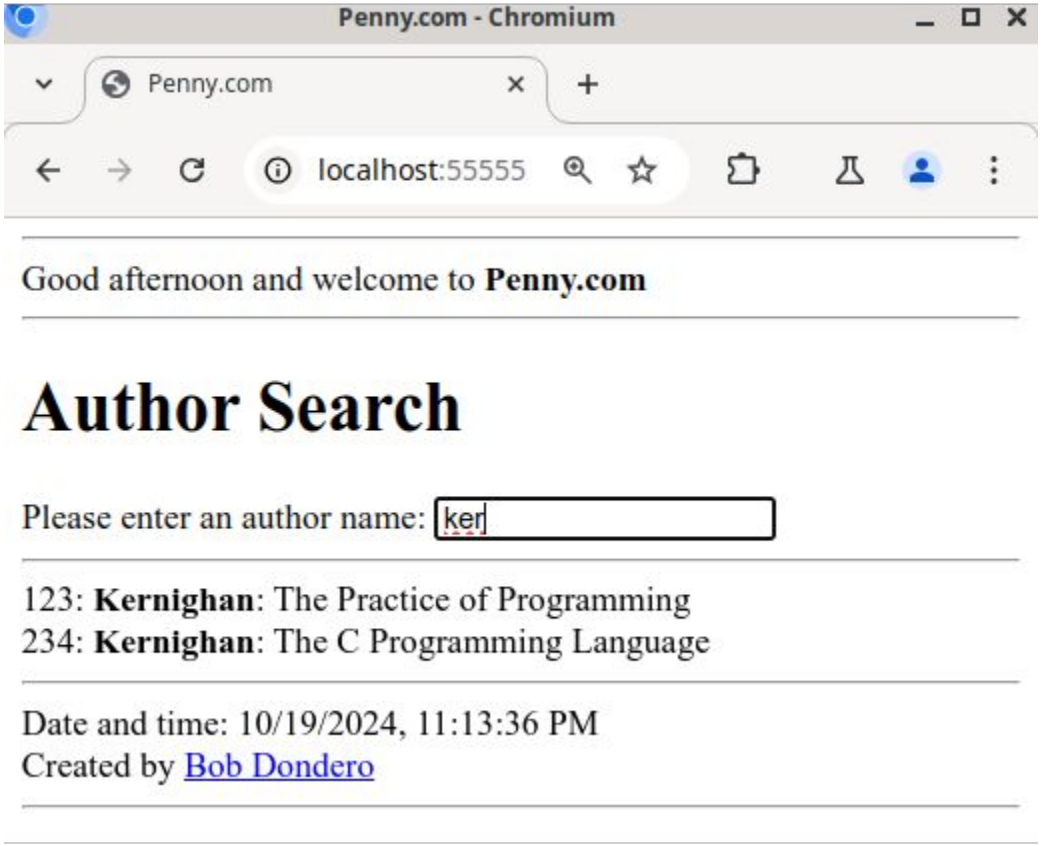
AJAX Revisited

- See **PennyXml** app



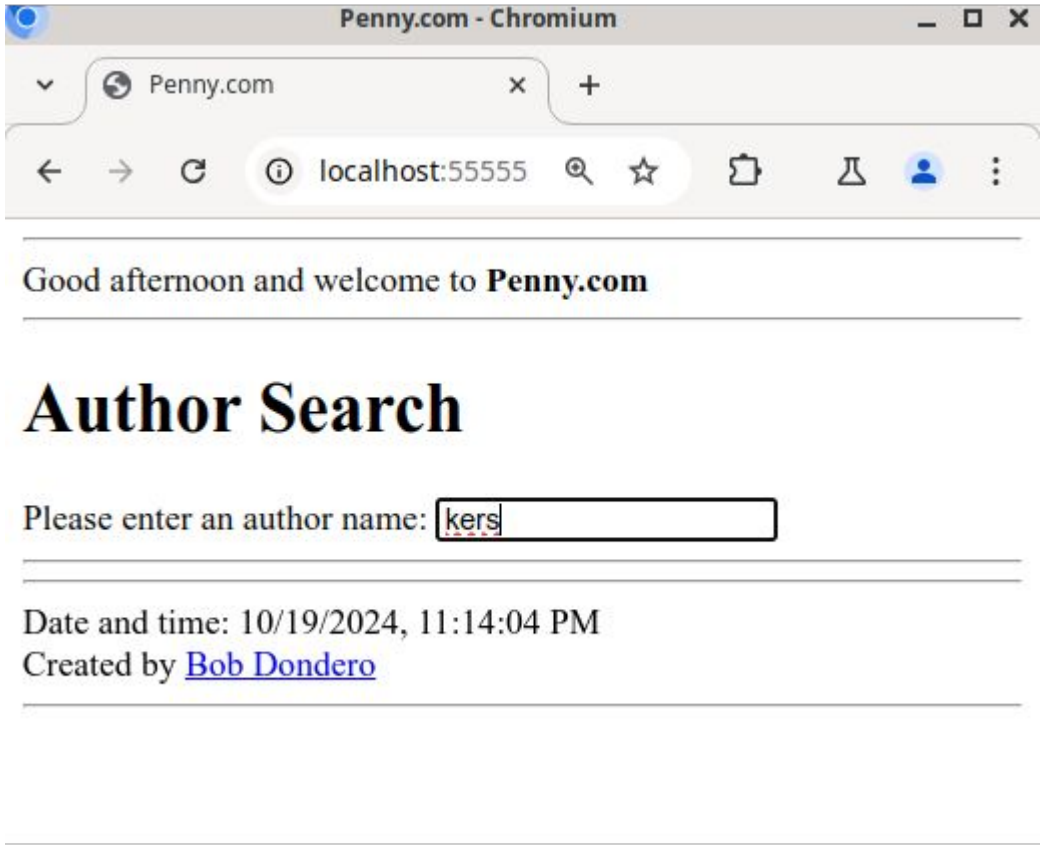
AJAX Revisited

- See **PennyXml** app (cont.)



AJAX Revisited

- See **PennyXml** app (cont.)



AJAX Revisited

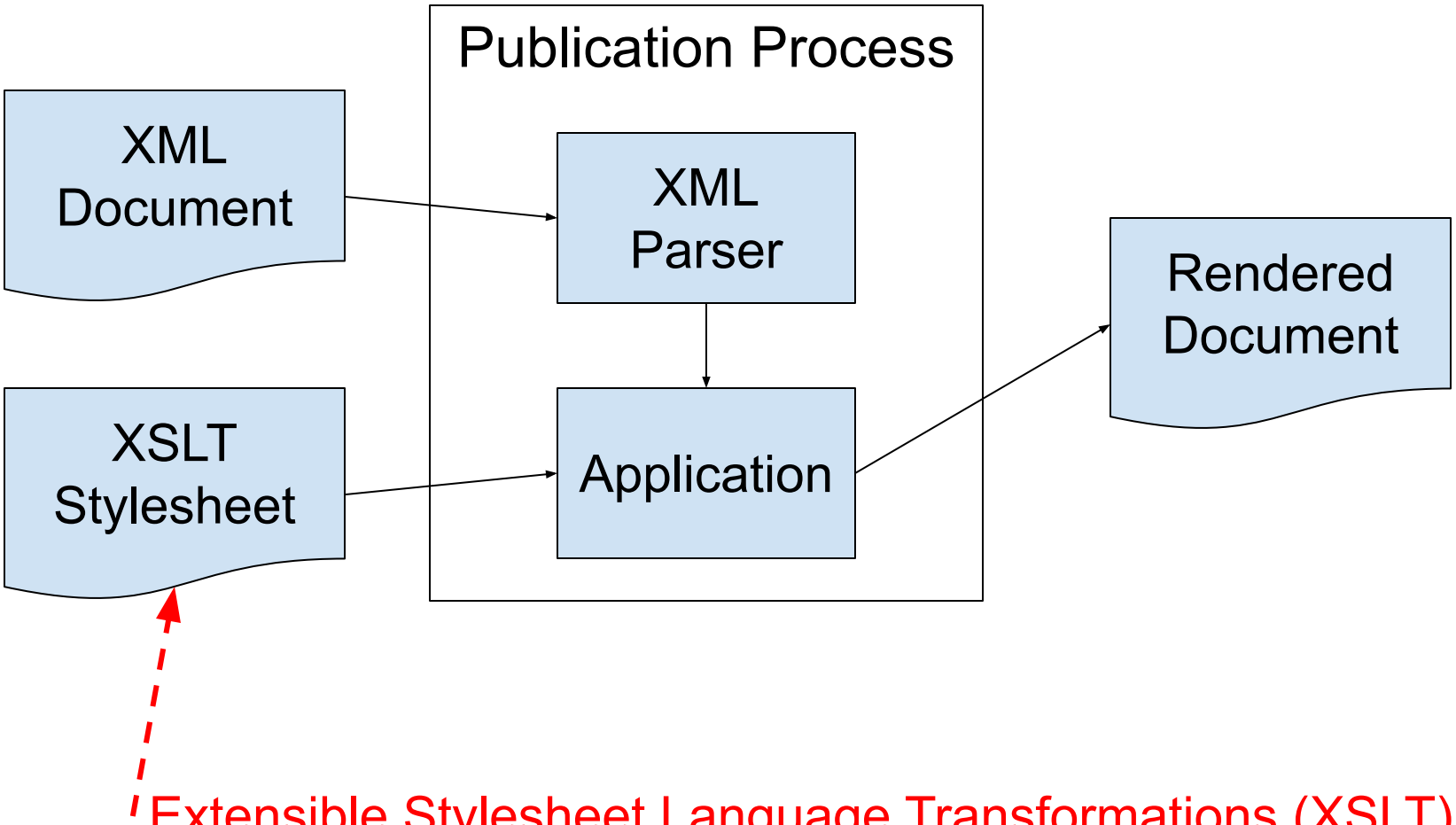
- See **PennyXml** app (cont.)
  - runserver.py
  - penny.sql
  - penny.sqlite
  - database.py
  - **penny.py**
  - **index.html**
    - AJAX automatically parses XML doc

Summary

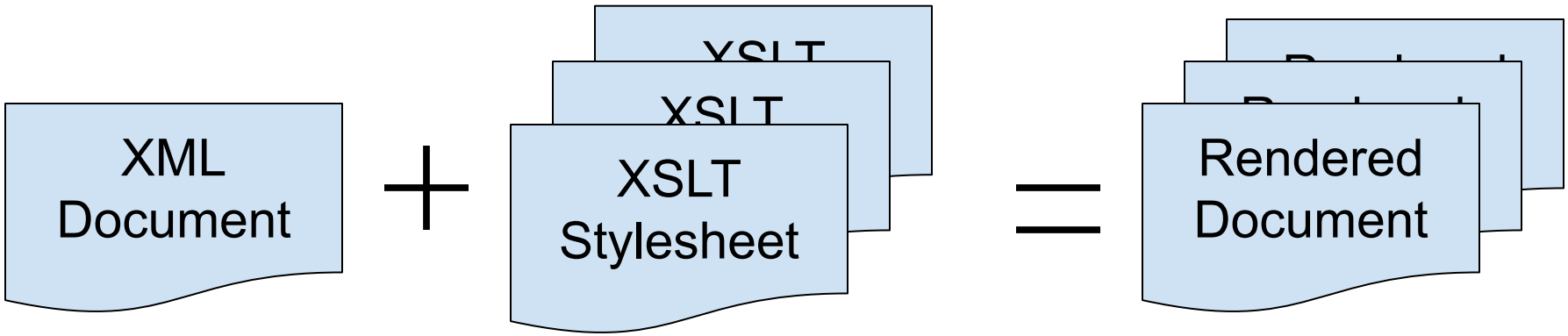
- We have covered:
  - XML
  - XML programming
  - AJAX revisited
- See also:
  - **Appendix 1**: XML for Publishing
  - **Appendix 2**: XML Programming: SAX
  - **Appendix 3**: XML Checking

Appendix 1:
XML for Publishing

XML for Publishing

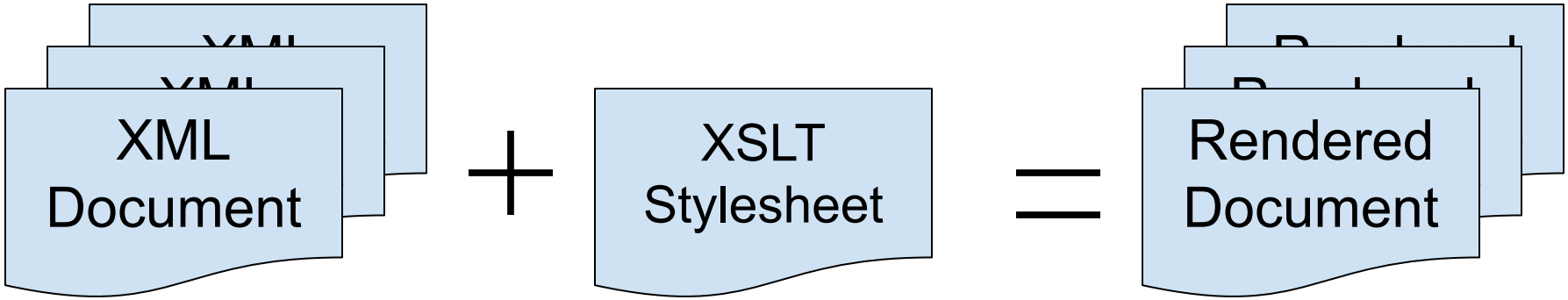


XML for Publishing



Same XML doc can be presented in multiple ways

XML for Publishing



Multiple XML docs can be presented in the same way

Appendix 2:
XML Programming: SAX

SAX Programming

- **Problem**
  - DOM trees can be very large
  - Sometimes you need only a small part
- **Solution**
  - Use SAX API instead of DOM API

SAX Programming

- A SAX parser:
  - Traverses given XML document
  - Calls methods (which you define) as it encounters each start tag, end tag, text, etc.

SAX Programming

- **writeauthorssax** programs
  - Write all authors in books.xml

SAX Programming

- See **writeauthorssax.py**

```
$ python writeauthorssax.py
Kernighan
Kernighan
Sedgewick
$
```

SAX Programming

- See **writeauthorssax.js**

```
$ node writeauthorssax.js
Kernighan
Kernighan
Sedgewick
$
```

Appendix 3:
XML Checking

XML Checking

- To run the example Python programs in this appendix:
  - `python –m pip install lxml`

XML Checking: Well-Formedness

- Computer science jargon…
  - An XML doc is ***well-formed*** iff it conforms to the XML specification

XML Checking: Well-Formedness

- See books.xml (revisited)
- See **booksmalformed.xml**
- See **checkxml.py**

XML Checking: Well-Formedness

```
$ python checkxml.py books.xml
The document is well-formed.
$ python checkxml.py booksmalformed.xml
mismatched tag: line 25, column 39
```

XML Checking: Validity

- ***DTD (Document Type Definition)***
  - An ISO standard
  - A DTD defines whether an XML doc is **valid**
- ***XSD (XML Schema Definition)***
  - A W3C standard
  - A XSD defines whether an XML doc is **valid**
- Others: ***RELAX NG***, ***Schematron***, ***DSDL***, …
  - See Wikipedia “XML” page
- Can use to define a comm protocol

XML Checking: Validity

- Computer science jargon…
  - A DTD or XSD defines a ***grammar***
  - The grammar defines a ***language***
    - A language is a set of documents
  - An XML document is ***valid*** with respect to a DTD or XSD iff it is an element of the language defined by that DTD or XSD

XML Checking: Validity

- See **books.dtd**
- See **booksusingdtd.xml**
- See **booksusingdtdinvalid.xml**
- See **checkxmlusingdtd.py**

XML Checking: Validity

```
$ python checkxmlusingdtd.py booksusingdtd.xml
The document is well-formed and valid.
$ python checkxmlusingdtd.py booksusingdtdinvalid.xml
Element book content does not follow the DTD,
expecting (isbn , author , title , price), got (isbn
author price title ), line 28, column 11 (<string>,
line 28)
$
```

XML Checking: Validity

- See **books.xsd**
- See **booksusingxsd.xml**
- See **booksusingxsdinvalid.xml**
- See **checkxmlusingxsd.py**

XML Checking: Validity

```
$ python checkxmlusingxsd.py booksusingxsd.xml
books.xsd
The document is well-formed and valid.
$ python checkxmlusingxsd.py booksusingxsdinvalid.xml
books.xsd
Element 'price': This element is not expected.
Expected is ( title ). (<string>, line 0)
$
```

XML Checking: Validity

- Advantages of validation via **DTDs**:
  - DTDs can be defined inline
  - DTDs are compact and readable
  - DTDs are widely supported

XML Checking: Validity

- Advantages of validation via **XSDs**:
  - XSDs support strong typing
    - Can specify that an element contains an integral number, real number, date, ...
  - XSDs provide natural way to map XML doc to typed objects
  - XSDs are written in XML; so can create/parse using XML tools
    - Can define a XSD, which defines a XSD, which ...