Csci 4131

Node.js + SQL + MySQL Revisited / Wrap-up

XML

Lecture 23, November 21st

Fall 2018

Dr. Dan Challou

Logistics

- Homework 6 (Login, Sessions, Database (SQL/MySQL) due next Tuesday 11/27 at 2:00 pm

Upcoming Reading/tutorials

XML – Sebesta, Chapter 7

Ajax Revisited – Sebesta Chapter 10

Tutorials:

https://www.w3schools.com/xml/default.asp

https://www.w3schools.com/xml/ajax_intro.asp

Then, PhP - Murach

Last Lecture

- Sessions
- SQL + MySQL + Node.js

Questions?

Lecture Path to HW 6

- Need to use the following technologies
  - ✓Node.js / javascript
  - ✓Express (Node.js framework)
  - ✓Sessions (enabled by express)
  - SQL + MySQL + Node.js

Today

- Node.js + SQL + MySQL Wrap-up (for now)
- XML

Review Exercise

- **Write an Express route named: logout that does the following:**
  - **Checks to see if a value in the session has been set and**
    - **If the value is not set, it prints the message:**

      "Session not started, can not logout" to the console
    - **Otherwise"**

      It destroys the session using the destroy() method

      It sends a response to the client with the message:

      "Session Complete"

SQL + MySQL + Node.js Revisited

- Start by understanding the interfaces with the queries we gave you:

- create_accounts_table.js
- create_places_table.js
- insert_into_accounts_table.js

Examples from Last Time Revisited

- Node.js + SQL + MySQL
  - Select Query
  - Insert Query

Another version of the Insert Query

XML – Extensible Markup Language

BOTTOM LINE UP FRONT: XML is a portable, widely supported, open (i.e., nonproprietary) language for data storage and exchange (i.e., transport)

XML vs HTML

- XML and HTML were designed with different goals:
  - XML was designed to transport and store data, with focus on what the data is
  - HTML was designed to display data, with focus on how the data looks
- HTML is about displaying information, while XML is about carrying information.

XML Basic Concepts 1

- XML is a meta language for defining any desired markup language for an application domain.
- –XML has no tags of its own, you define them!!!
- An XML-based markup language can be used to store and communicate information.
- Using XML, new tags and structures can be defined to describe the structure of the information to be stored and communicated.
- Newly created tags must adhere to the rules of XML specification.

XML Basic Concepts 2

- Information structured according to XML rules can be processed by computer programs in a meaningful way.
  - It can be parsed and manipulated according to given set of rules.
- XML is flexible - The structures used for information organization can be extended, and structures developed by different organizations can be combined.
- XML code is both data and the description of the DATA it holds.

Basic Concepts 3

- **Elements** are the basic building blocks of an XML document
- An **Element** contains a **tag**, and possibly some **attributes**. Every tag must have a closing tag.
- An Element may contain some **child elements**
- You invent your own tags in XML, there are no predefined tags
- A valid XML document must conform to certain structuring rules. The rules are defined either by a DTD (Document Type Definition)or an XML Schema.

Simple Example

```
<?xml version="1.0" encoding="UTF-8">
<person>
        <name>
                <firstname> Erik </firstname>
                <lastname> Kaler</lastname>
        </name>
        <height unit ="inches"> 71</height>
</person>
```

Document Prolog

<?xmlversion="1.0" encoding="UTF-8" **?>**

- The item above is a processing instruction. A document can have one or more processing instructions.
- A processing instruction is contained in brackets:

  <?PITarget … ?>

  - where the PITarget is a keyword for the document processing applications.

Document Prolog Continued

The Document prolog has three functions:

1. Indicate that this is an XML document.

2.Include some comments about the documents

- <!--This is a comment -->
- Warning: No nesting of comments. Do not use double hyphen within a comment

3.Includes some meta information about the contents of the document

Root element, and Element nesting

- Every document must have one and only one root element.
- In this example <person> … </person>is the root element.
- All elements have an opening tag and a closing tag.
- An element can have other nested elements:
  - Element **person** contains two elements: **name** and **height**.
  - Element **name** contains **firstname** and **lastname.**

Attributes and Values

- An element may contain one or more attributes.
- The typical use of an attribute is to represent some meta data for the data contained in the element.
- It defines properties or purpose of the content.
- in the previous example, element height contains attribute named units.
- <height unit= “inches”> 71 </height>

  attribute specifying that 71 is in inches
- A commonly used attribute is “id” as seen in XHTML, which must have a unique value in the document.

Exercise – Submit
Do your own version
put name and x.500 on it

- Define an XML Document that describes your favorite car. It should include the following prolog:

  <?xmlversion="1.0" encoding="UTF-8" **?>**

The root element should be named car, and the car should contain the name of the model and the year.

XML Namespaces

- XML namespaces provide a means for document authors to prevent naming collisions

- Each namespace prefix is bound to a uniform resource identifier (URI) that uniquely identifies the namespace
  - A URI is a series of characters that differentiate names
  - Document authors create their own namespace prefixes
  - Any name can be used as a namespace prefix, but the namespace prefix xml is reserved for use in XML standards

- To eliminate the need to place a namespace prefix in each element, authors can specify a default namespace for an element and its children
  - We declare a default namespace using keyword **xmlns** with a URI (Uniform Resource Identifier) as its value

- Document authors commonly use URLs (Uniform Resource Locators) for URIs, because domain names (e.g., deitel.com) in URLs must be unique

```
<?xml version = "1.0"?>

<!-- Fig. 15.5: namespace.xml -->
<!-- Demonstrating namespaces -->
<text:directory
   xmlns:text = "urn:deitel:textInfo"
   xmlns:image = "urn:deitel:imageInfo">

   <text:file filename = "book.xml">
      <text:description>A book list</text:description>
   </text:file>

   <image:file filename = "funny.jpg">
      <image:description>A funny picture</image:description>
      <image:size width = "200" height = "100" />
   </image:file>
</text:directory>
```

Fig. 15.5 | XML namespaces demonstration.

```
1  <?xml version = "1.0"?>
2
3  <!-- Fig. 15.6: defaultnamespace.xml -->
4  <!-- Using default namespaces -->
5  <directory xmlns = "urn:deitel:textInfo"
6     xmlns:image = "urn:deitel:imageInfo">
7
8     <file filename = "book.xml">
9        <description>A book list</description>
10    </file>
11
12    <image:file filename = "funny.jpg">
13       <image:description>A funny picture</image:description>
14       <image:size width = "200" height = "100" />
15    </image:file>
16 </directory>
```

Fig. 15.6 | Default namespace demonstration.

Validation

- Document Type Definitions
- XML Schema Documents