Web Data Interchange Formats

CSC 342 - Web Technologies

Web Data Exchange

- *Data exchange* is the process of transforming structured data from one format to another to facilitate data sharing between programs
- A *data exchange language* is a language that is capable of expressing general purpose data
- We will look at two data exchange languages that are commonly used to exchange data on the web:
  - Extensible Markup Language (XML)
  - JavaScript Object Notation (JSON)

Extensible Markup Language (XML)

- XML is a markup language that uses a textual data format to represent arbitrary data structures
- Schema systems exist for XML that enable XML based languages to be defined and validated
- The file extension for XML files is `.xml`
- The MIME type for XML text is `application/xml`

XML vs HTML

- XML was designed as a data exchange format
- HMTL was designed to display data to the user
- XML tags are not predefined
- XML was designed to be extensible

Basic XML Syntax Rules

- An XML document begins with an optional declaration, for example:

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  ```

- XML documents must have a root element that is the parent of all other elements
- XML elements must have a closing tag
- XML tags are case sensitive
- XML elements must be properly nested
- XML attributes must be quoted

XML Elements

- An XML element is a logical document component with the following basic syntax:

  `<element attribute="value">content</element>`

- Tags begin with < and end with >
  - *start-tag*: `<element>`
  - *end-tag*: `</element>`
  - *empty-element-tag*: `<element />`
- Attribute values must be in quoted (single or double quotes)

XML Element Naming Rules

- Element names are case sensitive
- Element names must start with a letter or an underscore
- Element names cannot start with the letters xml (in any combination of upper and lower case)
- Element names can contain letters, digits, hyphens, underscores, and periods
- Element names cannot contain spaces

XML Entities

- There are five pre-defined entity references
  - `&lt;` less than (<)
  - `&gt;` greater than (>)
  - `&amp;` ampersand (&)
  - `&apos;` apostophe (')
  - `&quot;` quotation mark (")

XML Comments

- Comments in XML begin with `<!–` and end with `–>`

```
<!-- This is a comment -->
```

- Two dashes – are not allowed in a comment
- Comments cannot be nested as a consequence of the previous rule

XML Namespaces

- XML elements are not predefined, so there is a chance that two different XML documents use the same element name
- XML Namespaces are a way to handle element name conflicts
- A namespace declaration has the following syntax
  `xmlns:prefix="URI"`
- Example:

```
<p:ul xmlns:p="https://example.com/p">
  <p:li>Item 1</p:li>
  <p:li>Item 2</p:li>
</p:list>
```

Accessing XML with JavaScript

- The XML DOM defines properties and methods for accessing and editing XML
- XML text data can be converted into an XML DOM object using the DOMParser object

```
var parser = new DOMParser();
x = parser.parseFromString(t, "text/xml");
// t is variable containing an XML string
```

Valid XML Documents

- A *well formed* XML document follows the XML syntax rules
- A *valid* XML document must be well formed and conform to a document type definition
- There are two main schema systems that can be used with XML
  - Document Type Definition (DTD)
  - XML Schema
- The purpose of a schema is to define the structure of an XML document including:
  - The elements and attributes that can appear in a document
  - The number and order of child elements
  - The data types for elements and attributes
  - The default and fixed values for elements and attributes

JavaScript Object Notation (JSON)

- JSON is designed to be a lightweight data exchange language
- JSON is data is plain text
- The file extension for JSON files is `.json`
- The MIME type for JSON text is `application/json`

JSON Syntax

- JSON syntax is similar to the syntax of defining literal objects in JavaScript:
  - Data is in name/value pairs of the form `"name":value`
  - Data is separated by commas
  - Curly braces hold objects
  - Square braces hold arrays

JSON Data Types

- **Number:** a signed decimal number
- **String:** a sequence of zero or more unicode characters delimited by by double quotes
- **Boolean:** a value of `true` or `false`
- **Array:** an ordered list of zero or more values separated by commas and delimited by square brackets
- **Object:** an unordered collection of name/value pairs where pairs are separated by commas and delimited by curly braces
- **Null:** the empty value indicated by the word `null`

Accessing JSON with JavaScript

- `JSON.parse()`: convert a JSON string into a JavaScript type
- `JSON.stringify():` convert a JavaScript object into JSON text
  - JavaScript Date objects are converted into strings
  - JavaScript functions are removed since they are not valid

XML vs JSON

- Similarities
  - "self describing" (human readable)
  - hierarchical (nested values)
  - can be parsed by many programming languages
  - can be fetched with `XMLHttpRequest`
- Differences
  - JSON does not use end tags
  - XML is more difficult to parse than JSON
  - JSON has an array type
  - JSON cannot be validated by a schema (yet)