

What is data?



Data Definition

- **Definition of data**
- 1: factual information (such as measurements or statistics) used as a ***basis for reasoning, discussion, or calculation***
- 2: information in digital form that ***can be transmitted or processed***
- 3: information output by a sensing device or organ that includes both useful and irrelevant or redundant information and ***must be processed to be meaningful***
- Definition provided by Merriam-Webster dictionary <https://www.merriam-webster.com/dictionary/data>



Data

- Data are the details
 - The bits
 - MAD202 <- this is a course code
 - WEB110 <- this is also a course code
 - MAD100 <- this is also a course code
- Bob
- Bill
- Gord
- George



These are names



Information

- Making sense of the data
- MAD202 and Bill together *means* that Bill is registered in MAD202
- Bill, Bob, Gord and George *means* we have four students in the course



..basis for reasoning, discussion, or calculation

- Used to make decision
- Do we have enough students registered?
- How much money did we earn in 2009?
- Which customers live in Ontario?



...can be transmitted or processed

- We often want to send data so that others can use this
- But they have to know what the data represents. For example:
- 1.21 <- data
- But what does it represent?
- The cost of an ice cream cone?
- The cost of a song on iTunes?
- The amount of money in your bank account?



...can be transmitted or processed

- So we need to format it – or label the data in a way that someone using it understands what is what
- Think of a table without headings - it is just a mishmash of data
 - We could probably guess at what the information represents
- We need to markup or establish what the data represents
 - Popular methods for marking up or identifying data that is sent via the web
 - JSON
 - XML



..must be processed to be meaningful

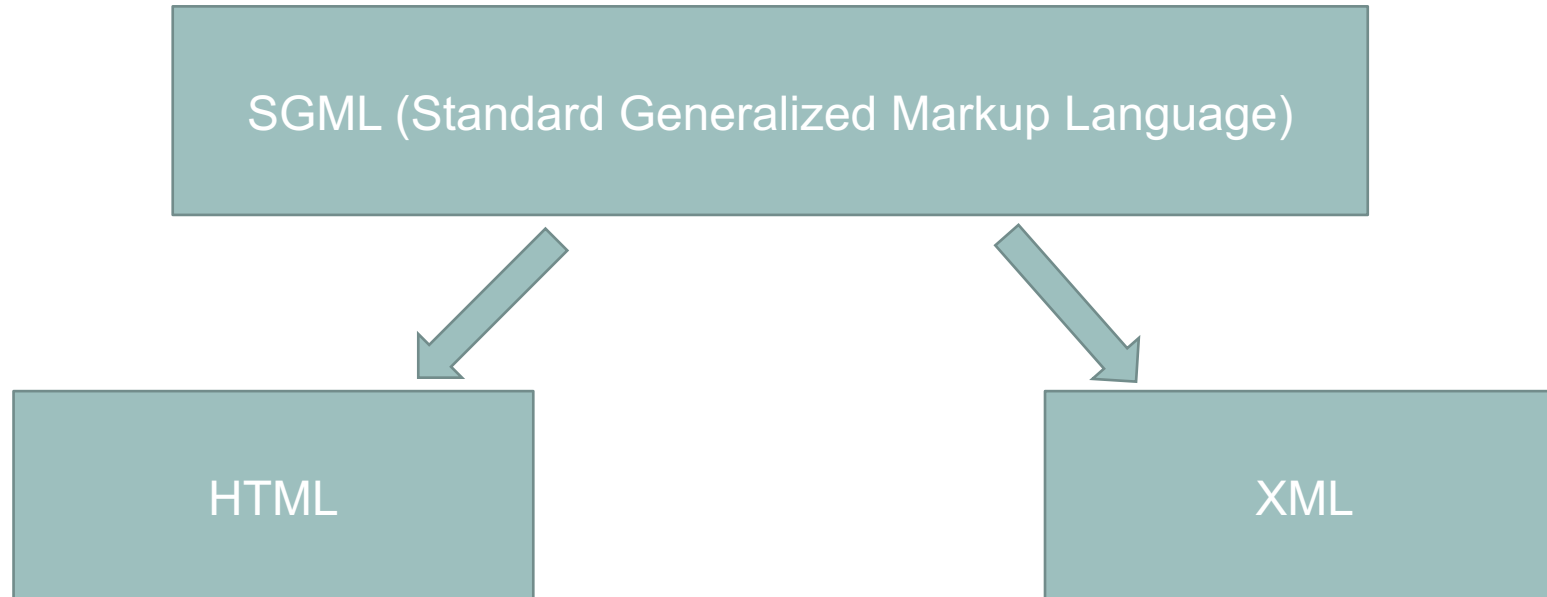
- We need to be able to do something with the data
- We identify it and then we take a dive into it
 - We sort it
 - We retrieve only what we want
 - We display portions or use it in calculations
 - We use it to get meaning from other pieces of information



What is XML?



What is XML?



What is XML?

XML (E**x**ensible **M**arkup **L**anguage)

- It is:

Technology concerned with the description and structuring of data

Not a language – but standard for creating languages that meet the XML criteria

Power comes from extensibility



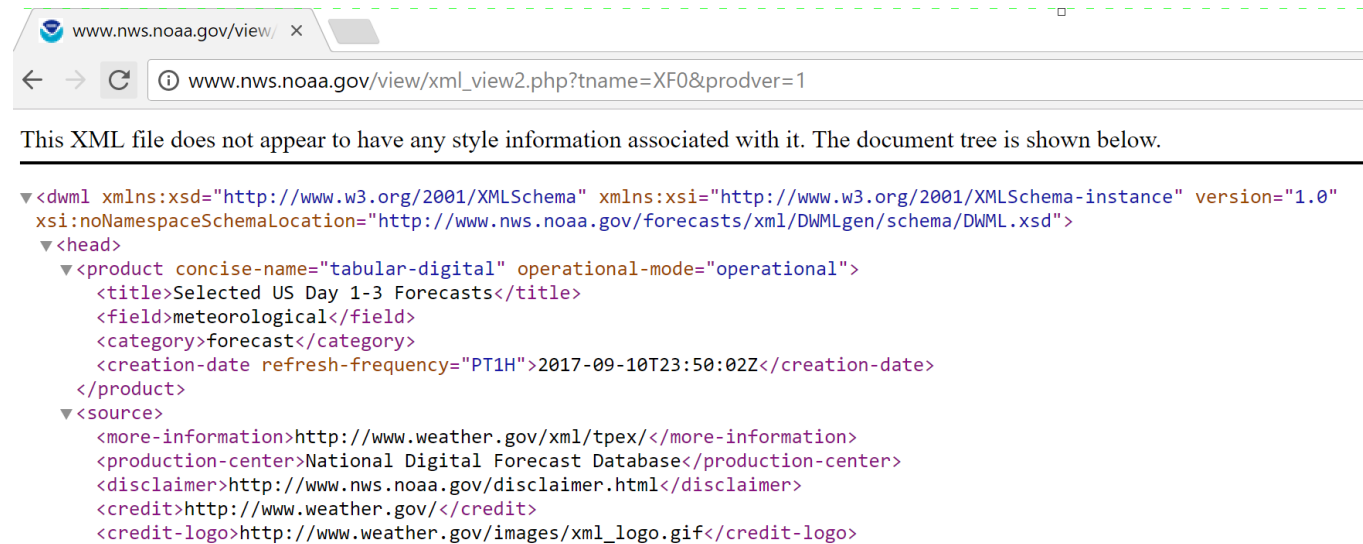
What is XML?

- Each XML file is a text file
- File extension - .xml
- The standard is developed and maintained by the World Wide Web Consortium



What is XML?

- What does it do?
- It doesn't do anything!

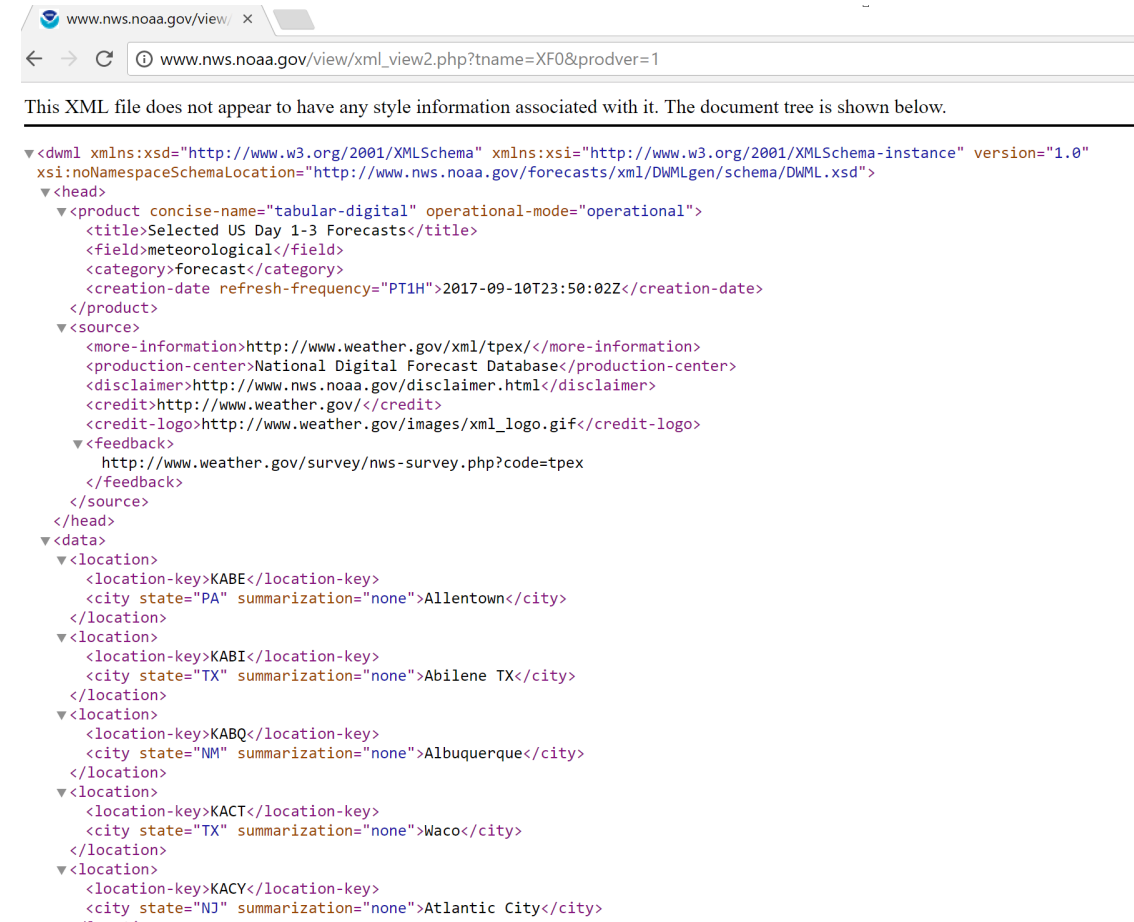


```
<dwml xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0"
xsi:noNamespaceSchemaLocation="http://www.nws.noaa.gov/forecasts/xml/DWMLgen/schema/DWML.xsd">
  <head>
    <product concise-name="tabular-digital" operational-mode="operational">
      <title>Selected US Day 1-3 Forecasts</title>
      <field>meteorological</field>
      <category>forecast</category>
      <creation-date refresh-frequency="PT1H">2017-09-10T23:50:02Z</creation-date>
    </product>
    <source>
      <more-information>http://www.weather.gov/xml/tpex/</more-information>
      <production-center>National Digital Forecast Database</production-center>
      <disclaimer>http://www.nws.noaa.gov/disclaimer.html</disclaimer>
      <credit>http://www.weather.gov/</credit>
      <credit-logo>http://www.weather.gov/images/xml_logo.gif</credit-logo>
    </source>
  </head>
</dwml>
```



What is XML?

- Where is it used?
 - Anywhere you need to exchange data:
 - Medical information
 - News Information
 - Weather Services
 - Financial Transactions
 - And more!



```
<?xml version="1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://www.nws.noaa.gov/forecasts/xml/DWMLgen/schema/DWML.xsd">
  <head>
    <product concise-name="tabular-digital" operational-mode="operational">
      <title>Selected US Day 1-3 Forecasts</title>
      <field>meteorological</field>
      <category>forecast</category>
      <creation-date refresh-frequency="PT1H">2017-09-10T23:50:02Z</creation-date>
    </product>
    <source>
      <more-information>http://www.weather.gov/xml/tpex/</more-information>
      <production-center>National Digital Forecast Database</production-center>
      <disclaimer>http://www.nws.noaa.gov/disclaimer.html</disclaimer>
      <credit>http://www.weather.gov/</credit>
      <credit-logo>http://www.weather.gov/images/xml_logo.gif</credit-logo>
    </source>
  </head>
  <data>
    <location>
      <location-key>KABE</location-key>
      <city state="PA" summarization="none">Allentown</city>
    </location>
    <location>
      <location-key>KABI</location-key>
      <city state="TX" summarization="none">Abilene TX</city>
    </location>
    <location>
      <location-key>KABQ</location-key>
      <city state="NM" summarization="none">Albuquerque</city>
    </location>
    <location>
      <location-key>KACT</location-key>
      <city state="TX" summarization="none">Waco</city>
    </location>
    <location>
      <location-key>KACY</location-key>
      <city state="NJ" summarization="none">Atlantic City</city>
    </location>
  </data>
</dwml>
```



Examples

- Let's Take a look at HTML and XML



How do I format XML?

The Rules



■ XML Document Structure

All XML documents must be “Well-formed”

The five (5) basic XML Document Rules:

1. Tags are case-sensitive
2. Opening tags must have closing tags
3. Tags must be properly nested
4. Attribute values must be quoted
5. Root “document” element must be *unique*, and is *required*.

Rule #1 -

- Tags are case sensitive

Dog is different from dog which is different from DOG

```
<Message>This is incorrect</message>
```

```
<message>This is correct</message>
```



Rule #2

Opening tags must have a closing tag

```
<firstName>Bruce  
<lastName>Wayne
```

```
<firstName>Bruce</firstName>  
<lastName>Wayne</lastName>
```



Rule #2

- You can include whitespace after the tag value (both opening and closing)

`<first >Clark</first >`

OR

`<first`

`>Clark</first>`



- Spaces are NOT allowed preceding the tag value

`< first>Clark< /first>`



Rule #3

- Tags must be properly nested

If you start element A, then start element B, you must first close element B before closing element A.

```
<book><title>This is not properly nested</book></title>
```

```
<book><title>This is properly nested.</title></book>
```



Rule #4

- Attribute values must be quoted

```
<note date=12/11/2007>  
  <to>Dave</to>  
  <from>John</from>  
</note>
```

```
<note date="12/11/2007">  
  <to>Dave</to>  
  <from>John</from>  
</note>
```

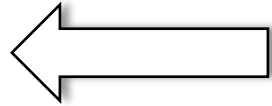


They can be single OR double – as long as they match each other

Rule #5

- Root element required and it must be unique – the beginning of your XML document.

<books>



The only piece of XML allowed outside the root element are comments and the processing instruction

<book>

<title>Cold Days</title>

<author>Jim Butcher</author>

...

</book>

</books>

XML Declaration

- Located at the very beginning – notes the version of XML you are using
- It is a processing instruction – not an element (no closing tag required)
- Before anything else
- Begin with the opening of the processing instruction(<? ?>)
- Version indicates the current version of XML to use (1.0)

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



XML Declaration

- Character encoding is the method used to represent the numbers in a character code digitally
- Can be included in your XML declaration
 - `<?xml version = "1.0" encoding = "UTF-8" ?>`
- If the encoding is NOT specified, UTF-8 or UTF-16 is assumed
- Only one encoding can be used for an entire XML document



XML Declaration

- The `standalone` attribute must be set to `yes` or `no` if used
- `Yes` specifies that the document exists entirely on its own without depending on any other files
- `No` indicates that the document *may* depend on a DTD (Document Type Definition)

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

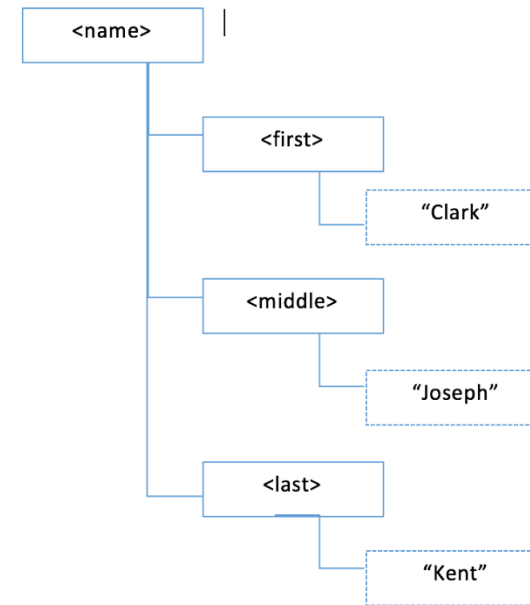
- `No` is assumed if not declared



XML hierarchy

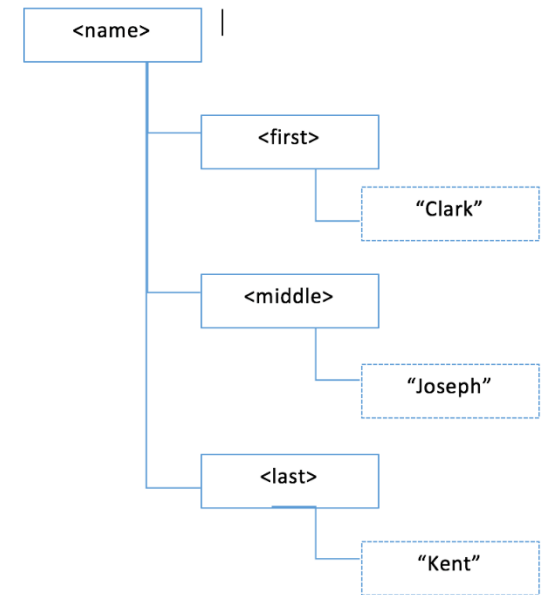
- XML groups information in hierarchies – the items relate to each other in parent/child and sibling/sibling relationships

```
<name>  
  <first>Clark</first>  
  <middle>Joseph</middle>  
  <last>Kent</last>  
</name>
```



XML hierarchy

- Also called a tree
 - Any part of the tree that contain children are called branches
 - Parts that have no children are called leaves
- Parent- child
 - `<name>` is the parent of `<first>`, `<middle>` & `<last>`
 - `<first>`, `<middle>`, `<last>` are siblings
 - The text (Clark) is a child of `<first>`



Writing XML - Elements

Must begin with a letter, or the dash (-) character, but no other numbers of punctuation – after that numbers, hyphens and decimals are allowed

No blank spaces

Can't begin with xml (either uppercase, lowercase or mixed)

Case Sensitive

Opening must match closing

Elements can contain other elements OR text (mixed content)

The name used should describe the elements purpose and its contents



Elements and Attributes



Non-Empty Elements

- Remember in XML you can make whatever elements you like
 - Case sensitive
 - Must begin with a slash or an underscore
 - No blank spaces
 - Can't begin with xml
 - Opening must match closing
 - Every element must be contained within the opening and closing root element

```
<name>Tiger</name>
```



Nesting Elements

- Remember elements can contain text or other elements
- Nesting elements allow you to establish hierarchical relationships with your data
- Each nested element must be completely nested in another

```
1  <?xml version="1.0" ?>
2  ▼ <superheroes>
3  ▼   <superhero>
4      <name>Superman</name>
5      <alterEgo>Clark Kent</alterEgo>
6   </superhero>
7
8  </superheroes>|
```



Attributes

Must begin with a letter or an underscore

No Spaces

Not begin with xml

Can only appear once within an element

Attributes are known as “name-value” pairs

Placed in the opening tag of an element

Generally considered meta data (data about the data)

Element can have multiple attributes as long as they are unique

MUST be in quotes – double or single



Attributes

- If a value contains double quotes – use single quotes to contain the value
 - `comments = 'She yelled, "Help Superman!"'`

```
1  <?xml version="1.0" ?>
2  ▼ <superheroes>
3  ▼   <superhero>
4       <name>Superman</name>
5       <alterEgo>Clark Kent</alterEgo>
6       <height units="feet">6.25</height>
7   </superhero>
8 </superheroes>
```



Why Use Attributes

- They tend to simplify your markup – you don't need to worry about proper nesting
- `<name nickname='Shiny John'></name>`
- Vs.
- `<name>`
- `<nickname> Shiny John</nickname>`
- `</name>`



Why use Attributes?

- Elements can make your markup bulky and difficult to read
- `<note>`
- `<type>Informational</type>`
- `This is a note.`
- `</note>`
- **Vs.**
- `<note type="informational">This is a note</note>`



When not to use Attributes?

- Attributes are unordered – so if order matters use elements
- Elements can be more complex
 - Attributes only allow for simple text as a value



Is this Valid?

```
<input checked = true>
```

Attributes must be quoted



Is this Valid?

```
<input checked = 'true'>
```

Attributes must be quoted



Is this Valid?

```
<input checked = "true">
```

Attributes must be quoted and the quotes must match



Is this Valid?

```
<input size='11px' size = '2em'>
```

No two attributes in a given element can share the same name



Is this Valid?

```
<name>Superman</ alias = "Clark Kent" name>
```

Attributes are attached to the start-tag



Empty Elements

- Do not have content
- The attributes store data for the element
- Has a single element that is self-closing

```
<main_image file="dog.jpg" />
```

OR

```
<main_image file="dog.jpg"></main_image>
```



Comments

- Useful to annotate XML documents
- As with HTML, comments are not parsed by the processor

```
<!-- update January 2014 -->
```

- No nesting of comments
- Can span multiple lines
- Hide sections during debugging (commenting out)
- Can contain spaces, text, elements
- Cannot nest a comment INSIDE an element
- Cannot use a – inside a comment



Is this Valid?

```
<!-- Did you know that Batman is -- Bruce Wayne -->
```

Can not have the double-dash inside a comment



Is this Valid?

```
<name <!-- this is the superhero's name --> >Superman</ name>
```

Comments cannot be nested inside of element tags



Special Characters

Work the same as in html – character references are used for non standard characters

There are only 5 predefined entities:

- `<`; represents "<"
- `>`; represents ">"
- `&`; represents "&"
- `'`; represents "'"
- `"`; represents "“"

Use within the text value of your XML document

- Any other entities that are used have to be pre-defined in a DTD



Text Characters

- Xml documents consist only of text characters
- Falls into three categories:
 - Parsed character data (PCDATA):
 - All the characters that XML treats as parts of markup tags
 - Character data (CDATA):
 - All that remains when pcddata is removed – this information is NOT parsed
 - White space



CDATA Section

- `<![CDATA[.....]]`
- Tells the parser to not parse the information
- Must be contained within the root element
- Can not have nested CDATA
- Special Symbols not required.
- Often used for large blocks of text or if you want to protect specific code from being processed by the XML processor

CDATA Section

```
<![CDATA[
```

```
Any information included here will be treated as text.
```

```
&amp; will display as &amp;
```

```
]]>
```

```
- <superheroes>
```

```
  - <superhero>
```

```
    <name>Superman</name>
```

```
    <alterEgo>Clark Kent</alterEgo>
```

```
    <height units="feet">6.25</height>
```

```
  </superhero>
```

```
    Any information included here will be treated as text. &amp; will display as &amp;
```

```
</superheroes>
```



Class Exercise

Build our first XML document



Sample Data

Type: Online (O) OR In-Store (I)

Date: four digit year (2016) - two digit month (01) - two digit day (13)

Phone: 10 digits - 5555555555

Name: Customer's Name

Email: email@domain.com

ID: Transaction ID (000001)

Product: Name of the product

Price: Cost of the item

ProdID: The ID for the product



Sample Data

- O
- 2016/01/11
- Henry Jones
- 555 888 9999
- hjonesr@marshallcollege.edu
- orderID 0000001
- leather journal
- \$23.991023983209

I
2016/01/07
Marcus Brody
555 888 2029
mbrody@marshallcollege.edu
orderID 0000002
calligraphy pen
\$51.991023987023
leather journal
\$23.991023983209

Class Exercise

