XML - Part 1

STAT 133

Gaston Sanchez

`github.com/ucb-stat133/stat133-fall-2016`

XML

XML & HTML

The goal of these slides is to give you a **crash introduction to XML and HTML** so you can get a good grasp of those formats for the following lectures

Datasets

You'll have some sort of (raw) data to work with

tabular

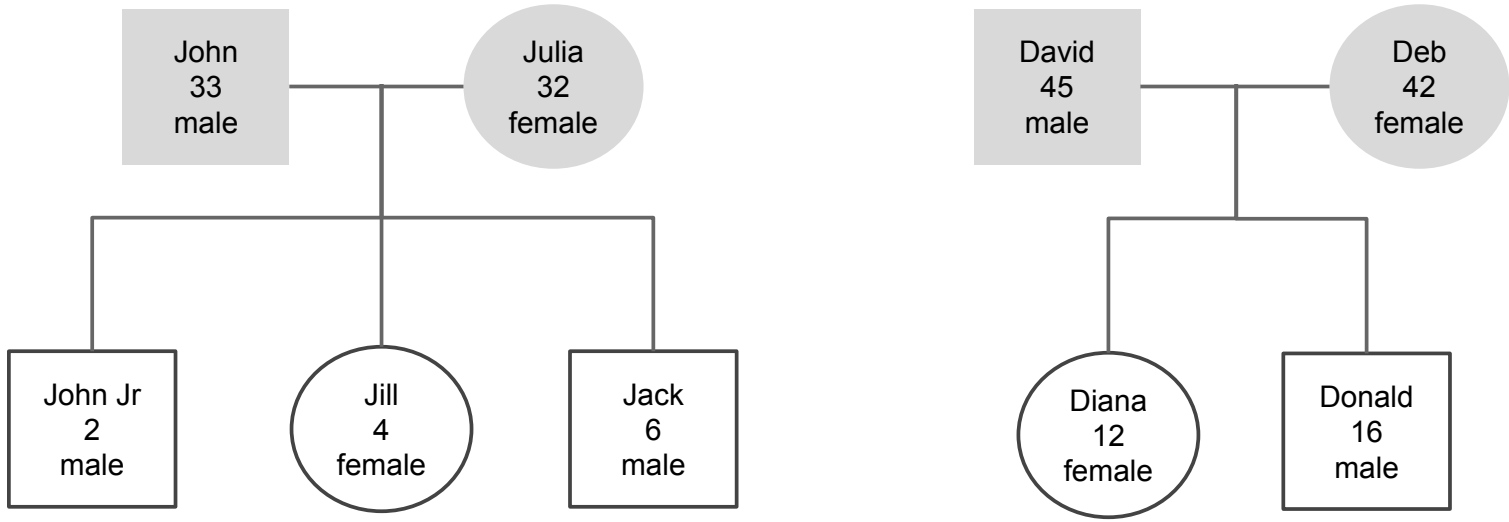
non-tabular

Motivation

Two main limitations of field-delimited files

- In plain text formats there is no information to describe the location of the data values
- There is no recognizable label for each data value within the file
- Serious limitations to store data with hierarchical structure

Hierarchical data



Hierarchical data

Field-delimited files have limitations with hierarchical data

		John	33	male
		Julia	32	female
John	Julia	Jack	6	male
John	Julia	Jill	4	female
John	Julia	John jnr	2	male
		David	45	male
		Debbie	42	female
David	Debbie	Donald	16	male
David	Debbie	Dianne	12	female

XML format

XML advantages

- XML is a storage format that is still based on plain text
- In XML formats every single value is distinctly labeled
- Moreover, every single value is self-described
- The information is organized in a much more sophisticated manner

Hierarchical data

An example of hierarchical data in XML

```
<family>
  <parent gender="male" name="John" age="33" />
  <parent gender="female" name="Julia" age="32" />
  <child gender="male" name="Jack" age="6" />
  <child gender="female" name="Jill" age="4" />
  <child gender="male" name="John jnr" age="2" />
</family>
<family>
  <parent gender="male" name="David" age="45" />
  <parent gender="female" name="Debbie" age="42" />
  <child gender="male" name="Donald" age="16" />
  <child gender="female" name="Dianne" age="12" />
</family>
```

XML and HTML

Why should you care about XML and HTML?

- Large amounts of data and information are stored, shared and distributed using HTML and XML-dialects
- They are widely adopted and used in many applications
- Working with data from the Web means dealing with HTML

XML

eXtensible Markup Language

Some Definitions

"XML is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable"

`http://en.wikipedia.org/wiki/XML`

"XML is a data description language used for describing data"

Paul Murrell

Introduction to Data Technologies

Some Definitions

"XML is a very general structure with which we can define any number of new formats to represent arbitrary data"

"XML is a standard for the semantic, hierarchical representation of data"

Deb Nolan & Duncan Temple Lang

XML and Web Technologies for Data Sciences with R

About XML

XML

XML stands for **eXtensible Markup Language**

Broadly speaking ...

XML provides a flexible framework to create formats for describing and representing data

Markups

Markup

A **markup** is a sequence of characters or other symbols inserted at certain places in a document to indicate either:

- how the content should be displayed when printed or in screen
- describe the document's structure

Markups

Markup Language

A markup language is a system for **annotating** (i.e. *marking*) a document in a way that the content is distinguished from its representation (eg LaTeX, PostScript, HTML, SVG)

LaTeX example

```
\documentclass{article}
\usepackage{graphicx}

\begin{document}

\title{Introduction to XML}
\author{First Last}
\maketitle

\section{Introduction}
Here is the text of your introduction.

\begin{equation}
    \label{simple_equation}
    \alpha = \sqrt{ \beta }
\end{equation}

\subsection{Subsection Heading Here}
Write your subsection text here.

\begin{figure}
    \centering
    \includegraphics[width=3.0in]{myfigure}
    \caption{Simulation Results}
    \label{simulationfigure}
\end{figure}

\end{document}
```

Markups

XML Markups

In XML (as well as in HTML) the marks (aka *tags*) are defined using angle brackets: <>

```
<mark>Text marked with special tag</mark>
```

Extensible

Extensible?

The concept of *extensibility* means that we can define our own marks, the order in which they occur, and how they should be processed. For example:

- `<my_mark>`
- `<awesome>`
- `<boring>`
- `<cool>`

About XML

XML is NOT

- a programming language
- a network transfer protocol
- a database

About XML

XML is

- more than a markup language
- a generic language that provides structure and syntax for representing any type of information
- a meta-language: it allows us to create or define other languages

XML Applications

Some XML dialects

- **KML** (*Keyhole Markup Language*) for describing geo-spatial information used in Google Earth, Google Maps, Google Sky
- **SVG** (*Scalable Vector Graphics*) for visual graphical displays of two-dimensional graphics with support for interactivity and animation
- **PMML** (*Predictive Model Markup Language*) for describing and exchanging models produced by data mining and machine learning algorithms

Keyhole Markup Language example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
  <name>New York City</name>
  <description>New York City</description>
  <Point>
    <coordinates>-74.006393,40.714172,0</coordinates>
  </Point>
</Placemark>
</Document>
</kml>
```

Scalable Vector Graphics example

```
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" />
</svg>


<svg width="400" height="110">
  <rect width="300" height="100" style="fill:rgb(0,0,255)" />
</svg>
```

Minimalist Example

XML Example

Ultra Simple XML

```
<movie>
  Good Will Hunting
</movie>
```

XML Example

Ultra Simple XML

```
<movie>
  Good Will Hunting
</movie>
```

- one single element *movie*
- start-tag: `<movie>`
- end-tag: `</movie>`
- content: `Good Will Hunting`

XML Example

Ultra Simple XML

```
<movie mins="126" lang="en">
  Good Will Hunting
</movie>
```

- xml elements can have **attributes**
- attributes: `mins` (minutes) and `lang` (language)
- attributes are *attached* to the element's start tag
- attribute values **must be quoted!**

XML Example

Minimalist XML

```
<movie mins="126" lang="en">
  <title>Good Will Hunting</title>
  <director>Gus Van Sant</director>
  <year>1998</year>
  <genre>drama</genre>
</movie>
```

- an xml element may contain other elements
- *movie* contains several elements: *title*, *director*, *year*, *genre*

XML Example

Simple XML

```
<movie mins="126" lang="en">
  <title>Good Will Hunting</title>
  <director>
    <first_name>Gus</first_name>
    <last_name>Van Sant</last_name>
  </director>
  <year>1998</year>
  <genre>drama</genre>
</movie>
```

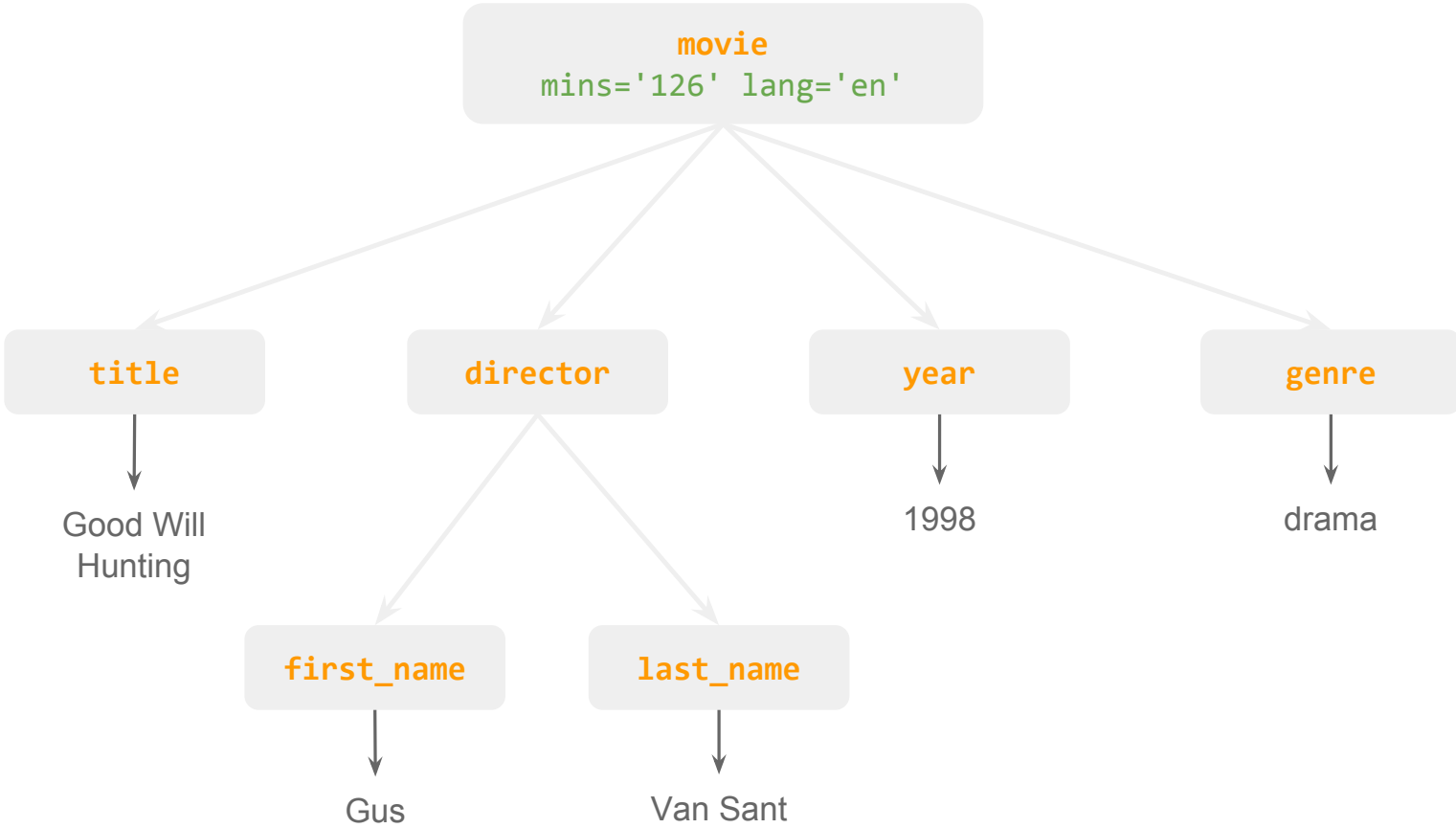
- Now *director* has two child elements: *first_name* and *last_name*

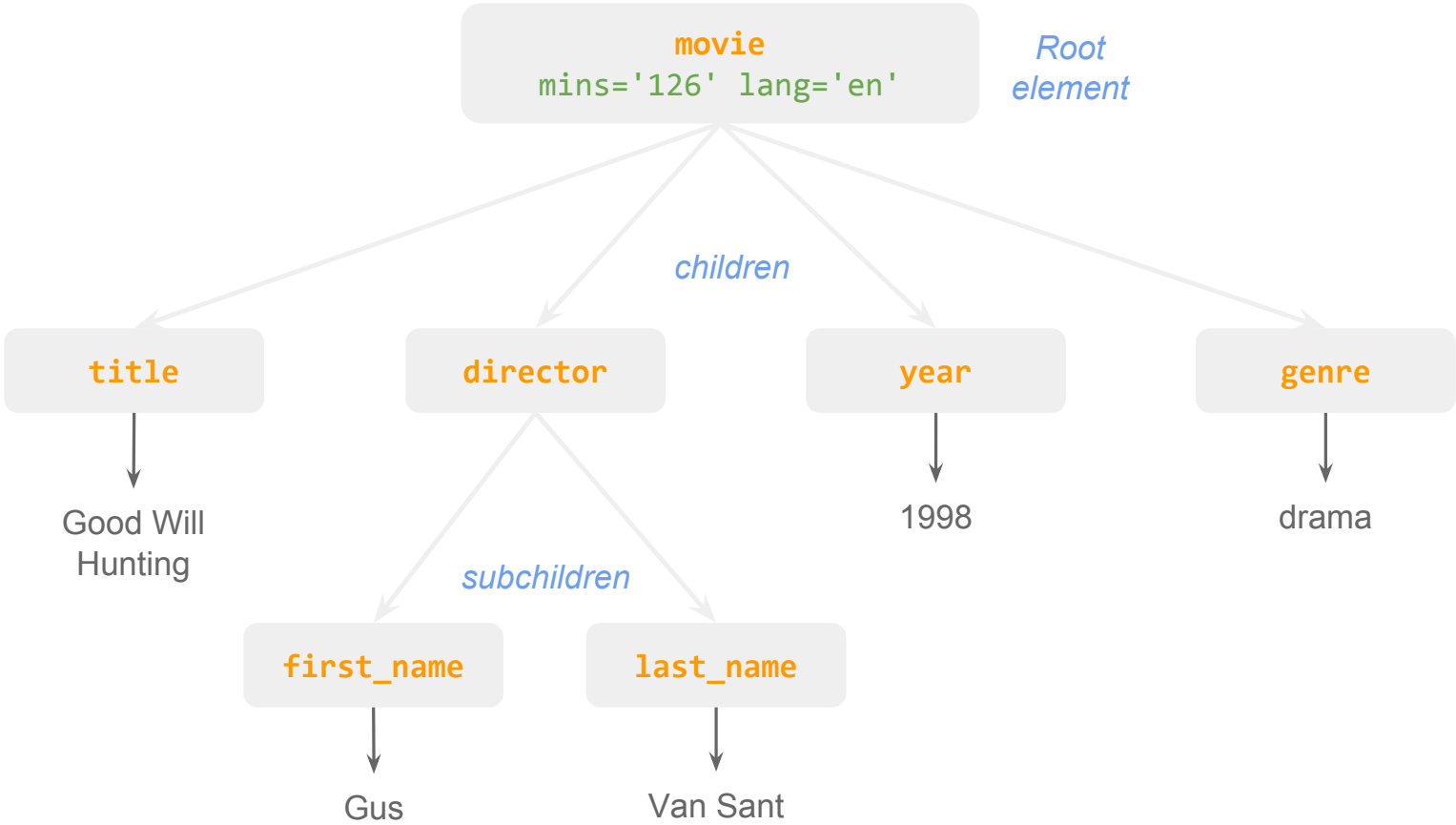
XML Hierarchy Structure

Conceptual XML

```
<Root>
  <child_1>...</child_1>
  <child_2>...</child_2>
    <subchild>...</subchild>
  <child_3>...</child_3>
</Root>
```

- An XML document can be represented with a **tree structure**
- An XML document must have **one single Root** element
- The `Root` may contain `child` elements
- A `child` element may contain `subchild` elements


movie
mins='126' lang='en'
title
director
year
genre
Good Will Hunting
1998
drama
first_name
last_name
Gus
Van Sant


movie
mins='126' lang='en'
Root element
children
title
director
year
genre
Good Will Hunting
1998
drama
subchildren
first_name
last_name
Gus
Van Sant

Well-Formedness

Well-formed XML

We say that an XML document is **well-formed** when it obeys the basic syntax rules of XML. Some of those rules are:

- one root element containing the rest of elements
- properly nested elements
- self-closing tags
- attributes appear in start-tags of elements
- attribute values must be quoted
- element names and attribute names are case sensitive

Well-Formedness

```
<movie mins="126" lang="en">
  <title>Good Will Hunting</title>
  <director>
    <first_name>Gus</first_name>
    <last_name>Van Sant</last_name>
  </director>
  <year>1998</year>
  <genre>drama</genre>
</movie>
```

Well-Formedness

Importance of Well-formed XML

Not well-formed XML documents produce potentially fatal errors or warnings when parsed.

Documents may be well-formed but not valid. Well-formed just guarantees that the document meets the basic XML structure, not that the content is valid.

Additional XML Elements

Some Additional Elements

```
<?xml version="1.0"? encoding="UTF-8" ?>
<![CDATA[ a > 5 & b < 10 ]]>
<?GS print(format = TRUE)>
<!DOCTYPE Movie>
<!-- This is a commet -->
<movie mins="126" lang="en">
  <title>Good Will Hunting</title>
  <director>
    <first_name>Gus</first_name>
    <last_name>Van Sant</last_name>
  </director>
  <year>1998</year>
  <genre>drama</genre>
</movie>
```

Additional Optional XML Elements

Markup	Description
`<?xml ?>`	XML Declaration *identifies content as an XML document*
`<?PI ?>`	Processing Instruction *processing instructions passed to application* `PI`
`<!DOCTYPE >`	Document-type Declaration *defines the structure of an XML document*
`<![CDATA[ ]]>`	CDATA Character Data *anything inside a CDATA is ignored by the parser*
`<!-- -->`	Comment *for writing comments*

Document-Type Declaration

The Document-type Declaration identifies the **type** of the document. The *type* indicates the structure of a **valid** document:

- what elements are allowed to be present
- how elements can be combined
- how elements must be ordered

Basically, the DTD specifies what the format allows to do.

Wrapping Up

About XML

- designed to store and transfer data
- designed to be self-descriptive
- tags are not predefined and can be extended

Characteristics of XML

XML is

- a generic language that provides structure and syntax for many markup dialects
- is a syntax or format for defining markup languages
- a standard for the semantic, hierarchical representation of data
- provides a general approach for representing all types of information dialects

XML document example

Simple XML

```
<?xml version="1.0"?>
<!DOCTYPE movies>
<movie mins="126" lang="en">
  <!-- this is a comment -->
  <title>Good Will Hunting</title>
  <director>
    <first_name>Gus</first_name>
    <last_name>Van Sant</last_name>
  </director>
  <year>1998</year>
  <genre>drama</genre>
</movie>
```

XML Tree Structure

Each Node can have:

- a Name
- any number of attributes
- optional content
- other nested elements

Traversing the tree

There's a **unique** path from the root node to any given node

HTML

HTML

About HTML

- HyperText Markup Language
- standard markup language used to create web pages
- HTML describes the structure of a website semantically along with cues for presentation
- Web browsers can read HTML files and render them into visible or audible web pages

Hello World example

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

HTML

- Open a new text file
- Add some HTML content (e.g. hello world example)
- Save your file with extension `.html`
- Click on your html file
- Should be displayed in your browser

Header Element

Header of the HTML document: is declared with the tag `<head>...</head>`

```
<head>
  <title>The Title</title>
</head>
```

Headings

HTML headings are defined with the `<h1>, <h2>, ... <h6>` tags:

```
<h1>Heading level 1</h1>
<h2>Heading level 2</h2>
<h3>Heading level 3</h3>
<h4>Heading level 4</h4>
<h5>Heading level 5</h5>
<h6>Heading level 6</h6>
```

Paragraphs

Paragraphs are defined with the <p> tag:

```
<p>This is the first paragraph</p>

<p>
  This is the second paragraph.
  The quick brown fox jumps over the lazy dog.
</p>
```

Links and comments

Links require the anchor tag `<a>` and the attribute `href=`

```
<a href="https://www.wikipedia.org/">A link to Wikipedia!</a>
```

Comments:

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

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

Images

Images are included with the `<img>` tag and the attribtue `src=`:

```
<img src="image.gif">
```

Image with a link:

```
<a href="http://example.org">
  <img src="image.gif" alt="descriptive text">
</a>
```

HTML Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <!-- this is a commetn -->
  <body>
    <h1>Heading level 1</h1>
    <h2>Heading level 2</h2>
    <h3>Heading level 3</h3>
    <h4>Heading level 4</h4>
    <h5>Heading level 5</h5>
    <h6>Heading level 6</h6>

    <p>Hello world!</p>
    <img src="http://www.r-statistics.com/wp-content/uploads/2013/05/R_logo-e136

    <a href="https://www.r-project.org/">This is a link</a>
  </body>
</html>
```

Some References

- XML Files website (http://www.xmlfiles.com)
  by Jan Egil Refsnes
- XML in a Nutshell
  by Elliotte Rusty Harold; W. Scott Means
- XML Tutorial (http://www.w3schools.com/xml/default.asp)
  by w3schools
- Introduction to Data Technologies
  by Paul Murrell
- XML and Web Technologies for Data Sciences with R
  by Deb Nolan and Duncan Temple Lang