

**X****M****L**

# Data Management

dott. Chris Mair  
2009/2010 @unibz

I – Introduction

last modified 2009-10-07

# So... why XML?

let's have a look at...

- an OpenOffice.org `.odt` file  
(*“Open Document Format”*)
- a Microsoft `.docx` file  
(*“Office Open XML”*)
- the Apple `.key` file you're reading
- a config file of some tool on my notebook

**See some pattern?**

# Ok, some context!

- early-'80 SGML (ANSI)
- early-'90s HTML (CERN, Tim Berners-Lee)
- mid-'90s XML (W3C)
- '98 XML 1.0 is a W3C recommendation:

<http://www.w3.org/TR/REC-xml>

# Sample document

```
<?xml version="1.0"?>
<!DOCTYPE telegram SYSTEM "telegram.dtd">
<telegram pri="important">
  <to>Sarah Bellum</to>
  <from>Colonel Timeslip</from>
  <subject>Robot-sitting instructions</subject>
  <graphic fileref="figs/me.eps"/>
  <message>Thanks for watching my robot pal
    <name>Zonky</name> while I'm away.
    He needs to be recharged <emphasis>twice a
    day</emphasis> and if he starts to get cranky,
    give him a quart of oil. I'll be back soon,
    after I've tracked down that evil
    mastermind <villain>Dr. Indigo Riceway</villain>.
  </message>
</telegram>
```

source: "Learning XML"

# Anatomy of a document

XML declaration  
(optional)

```
<?xml version="1.0"?>
<!DOCTYPE telegram SYSTEM "telegram.dtd">
<telegram pri="important">
  <to>Sarah Bellum</to>
  <from>Colonel Timeslip</from>
  <subject>Robot-sitting instructions</subject>
  <graphic fileref="figs/me.eps"/>
  <message>Thanks for watching my robot pal
    <name>Zonky</name> while I'm away.
    He needs to be recharged <emphasis>twice a
    day</emphasis> and if he starts to get cranky,
    give him a quart of oil. I'll be back soon,
    after I've tracked down that evil
    mastermind <villain>Dr. Indigo Riceway</villain>.
  </message>
</telegram>
```

source: "Learning XML"

# Anatomy of a document

document type  
declaration  
(optional)

```
<?xml version="1.0"?>  
<!DOCTYPE telegram SYSTEM "telegram.dtd">  
<telegram pri="important">  
  <to>Sarah Bellum</to>  
  <from>Colonel Timeslip</from>  
  <subject>Robot-sitting instructions</subject>  
  <graphic fileref="figs/me.eps"/>  
  <message>Thanks for watching my robot pal  
    <name>Zonky</name> while I'm away.  
    He needs to be recharged <emphasis>twice a  
    day</emphasis> and if he starts to get cranky,  
    give him a quart of oil. I'll be back soon,  
    after I've tracked down that evil  
    mastermind <villain>Dr. Indigo Riceway</villain>.  
  </message>  
</telegram>
```

source: "Learning XML"

# Anatomy of a document

document element  
(mandatory)

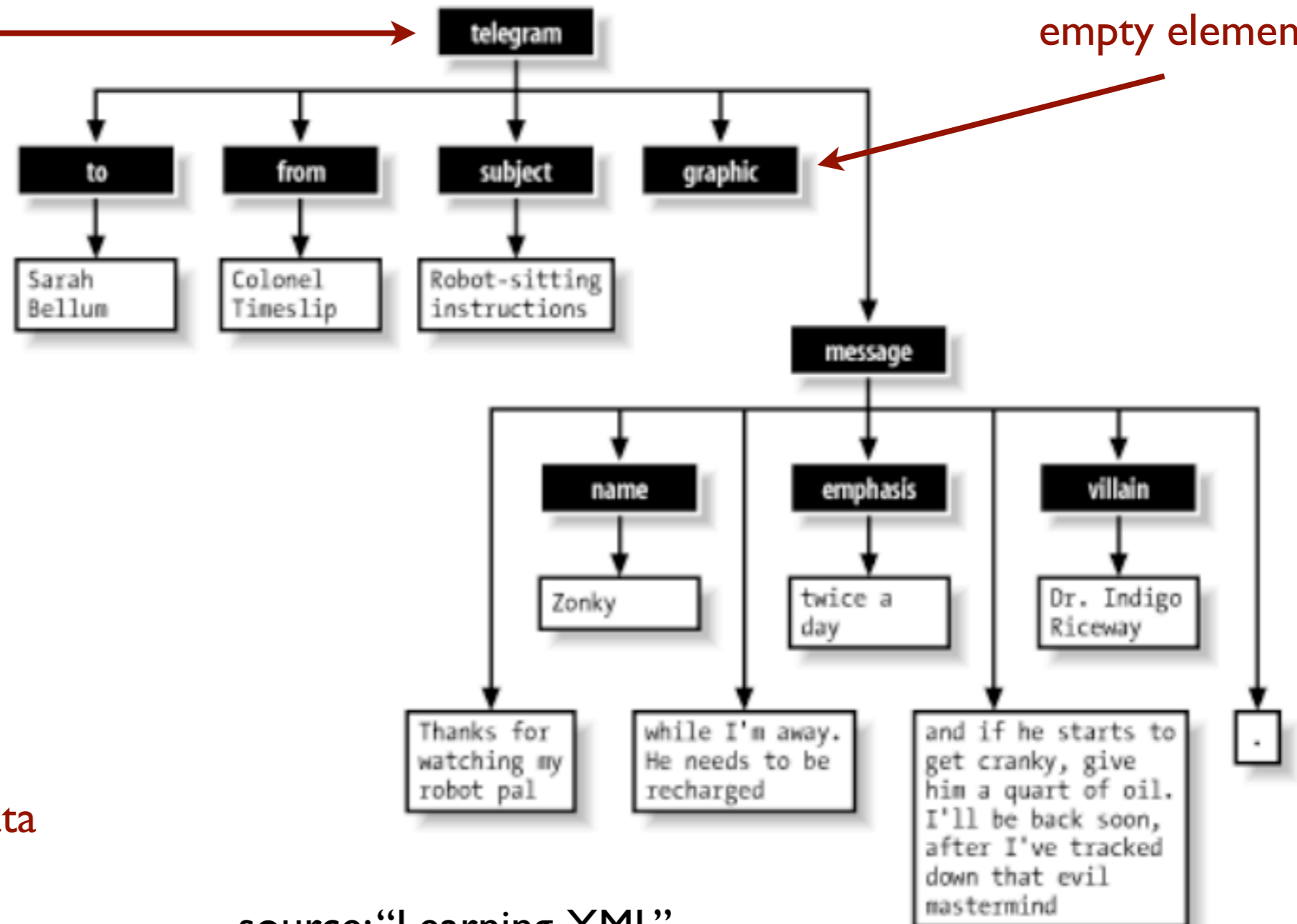
```
<?xml version="1.0"?>
<!DOCTYPE telegram SYSTEM "telegram.dtd">
<telegram pri="important">
  <to>Sarah Bellum</to>
  <from>Colonel Timeslip</from>
  <subject>Robot-sitting instructions</subject>
  <graphic fileref="figs/me.eps"/>
  <message>Thanks for watching my robot pal
    <name>Zonky</name> while I'm away.
    He needs to be recharged <emphasis>twice a
    day</emphasis> and if he starts to get cranky,
    give him a quart of oil. I'll be back soon,
    after I've tracked down that evil
    mastermind <villain>Dr. Indigo Riceway</villain>.
  </message>
</telegram>
```

source: "Learning XML"

# Tree representation

document  
element  
(root of tree)

empty element

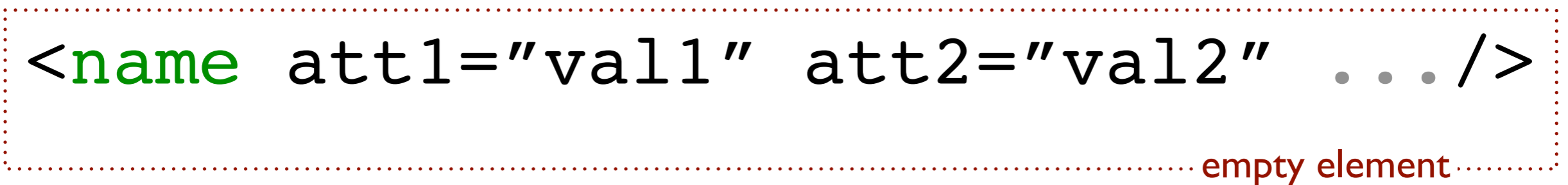
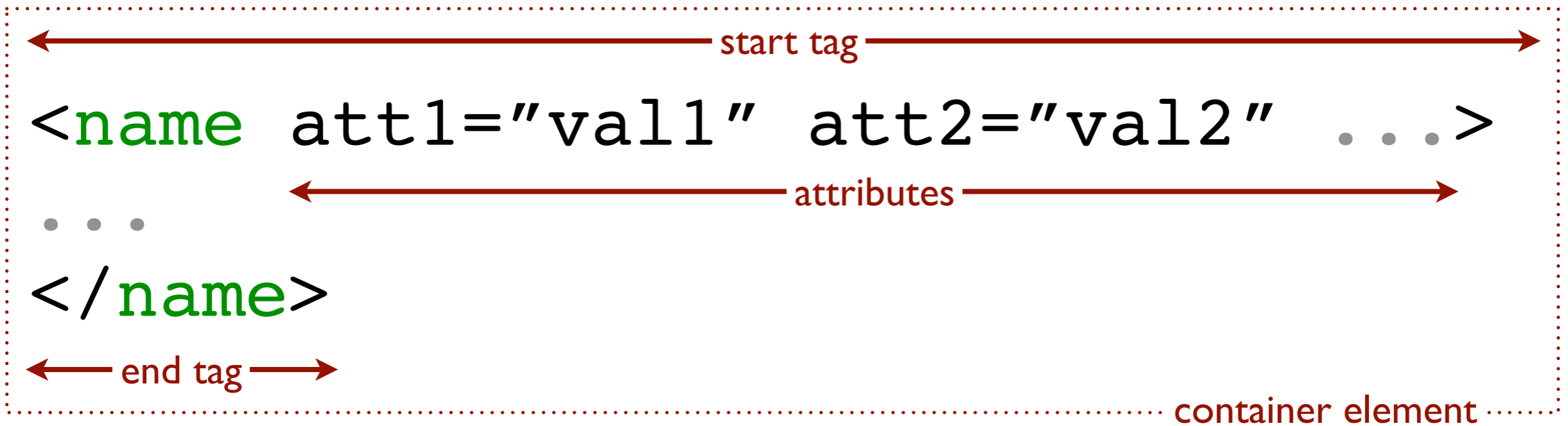


element

character data

source: "Learning XML"

# Anatomy of an element



# XML markup rules

- get the details from “*Learning XML - Chapter 2: Markup and Core Concepts*”
- the formal grammar is available from the W3C at:  
<http://www.w3.org/TR/REC-xml>
- documents that follow these rules are set to be *well-formed*

# Checking well-formedness

- in the lab we're going to use Apache *Xerces* as a parser to check XML documents for well-formedness:

```
$ grep "from" telegram.xml
```

```
<frm>Colonel Timeslip</from>
```

```
$ xerces dom.Counter telegram.xml
```

```
[Fatal Error] telegram.xml:5:26: The element type  
"frm" must be terminated by the matching end-tag  
"</frm>".
```

- see <http://xerces.apache.org/>

# What's next? Schemas!

- XML documents can be constraint to satisfy a given schema (there is an obvious analogy to the world of relational databases where data must follow the schema given by the table definitions)
- an XML document is said to be *valid* if it satisfies a given schema – well-formedness is a necessary condition for validity