The Idea of Markup

- **The following features of Values:**
  - Easy to read for humans
  - Easy to use
  - Easy to Use for Computer
  - Easy to debug
  - Easy to modify suitably for any industry or domain
  - Works with all leading programming languages, database and formats such as spredsheets and drawing.

Organising Information in XML

- Desiging an XML document is similar to designing a database table.
- This process break down into three steps:
  - **Classifiying information as per its importance**
  - **Adding the details**
  - **Transforming information into XML format**
    - **Identifying elements**
    - **Identifying attributes**

Organising Information in XML

- **Classifiying information as per its importance for BOOK**
  - Titler
  - Author
  - Publication
  - Price
  - Publishing Year
  - Reprint number
  - Edition number
  - Book Website

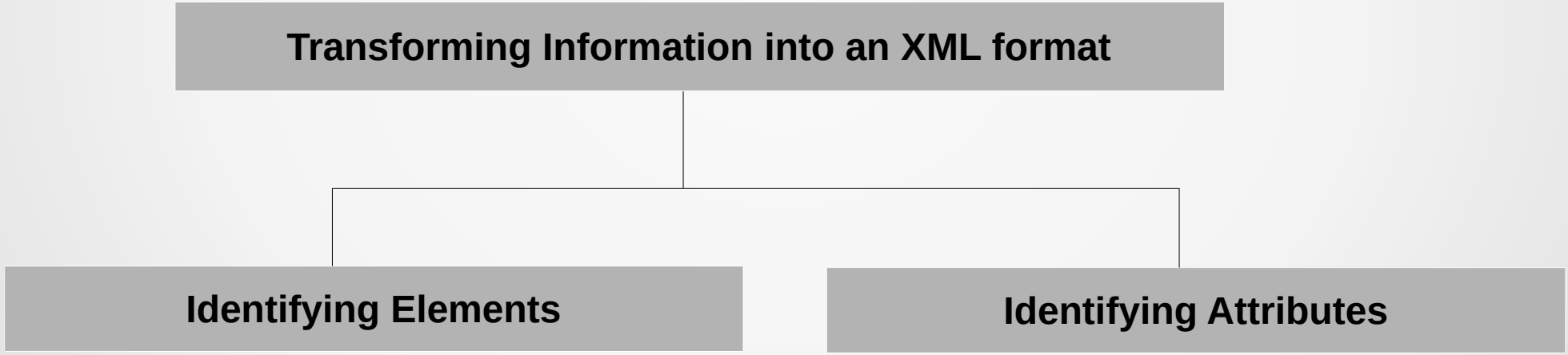
Organising Information in XML

- **Adding the details**

Primary Info	Details we want to capture	Details we can ignore
Title	Main Title Sub - Title	-
Autor	First Name Last Name	Full Name Affiliations
Publication	Name of Publisher	Full Address
Price	In local Currency	In more Currency
Edition	Number	-
Book Website	URL	-

Organising Information in XML

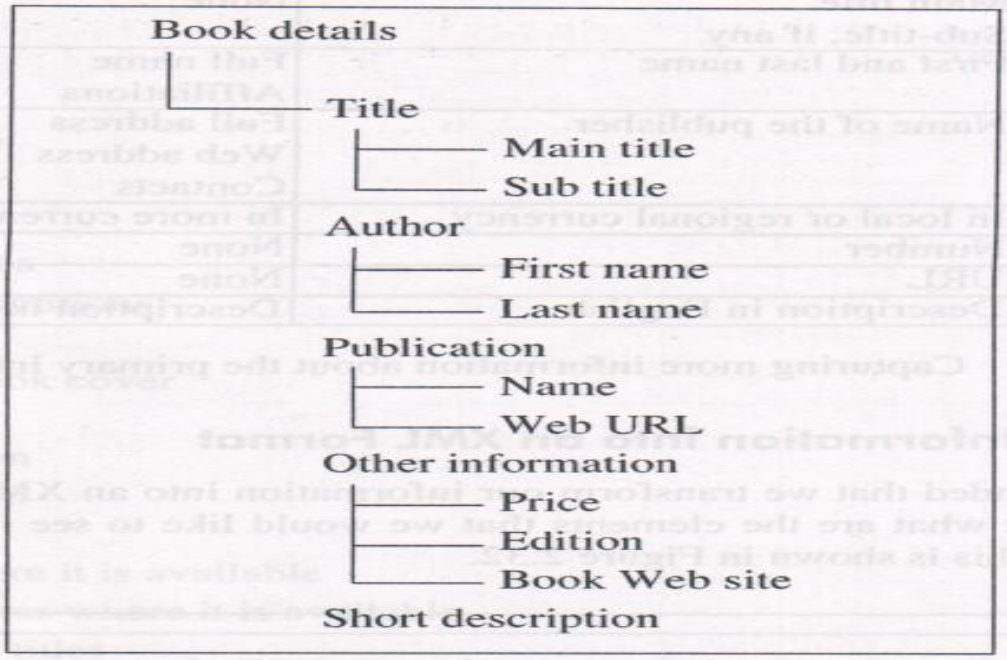
- **Transforming information into XML format**
  - **Identifying elements**
  - **Identifying attributes**



Organising Information in XML

- **Identifying Elements**

Figure 2.33 depicts the hierarchy of elements for our books example.



Organising Information in XML

- **Identifying Elements**

is to transform the visual form of the hierarchy into an XML-like syntax. The resulting structure Figure 2.34.

```
<BOOK_DETAILS>
    <TITLE>
        <MAIN_TITLE> </MAIN_TITLE>
        <SUB_TITLE> </SUB_TITLE>
    </TITLE>
    <AUTHOR>
        <FIRST_NAME> </FIRST_NAME>
        <LAST_NAME> </LAST_NAME>
    </AUTHOR>
    <PUBLICATION>
        <NAME> </NAME>
        <WEB_URL> </WEB_URL>
    </PUBLICATION>
    <OTHER_INFO>
        <PRICE> </PRICE>
        <EDITION> </EDITION>
        <BOOK_WEB_SITE> </BOOK_WEB_SITE>
    </OTHER_INFO>
    <SHORT_DESC> </SHORT_DESC>
</BOOK_DETAILS>
```

Organising Information in XML

- **Identifying Attributes**

ments, for instance, that of the TITLE. This approach is illustrated in Figure 2.38.

```
<BOOK_DETAILS>
    <TITLE category = "…">
        <MAIN_TITLE> </MAIN_TITLE>
        <SUB_TITLE> </SUB_TITLE>
    </TITLE>
    <AUTHOR>
        <FIRST_NAME> </FIRST_NAME>
        <LAST_NAME> </LAST_NAME>
    </AUTHOR>
    <PUBLICATION>
        <NAME> </NAME>
        <WEB_URL> </WEB_URL>
    </PUBLICATION>
    <OTHER_INFO>
        <PRICE> </PRICE>
        <EDITION> </EDITION>
        <BOOK_WEB_SITE> </BOOK_WEB_SITE>
    </OTHER_INFO>
    <SHORT_DESC> </SHORT_DESC>
</BOOK_DETAILS>
```

Figure 2.38 Adding the category attribute to a book

Creating Well – Formed XML Documents

- **The <?xml> tag**
  - This tag identifies **our document as an XML Document.**
  - It **must be first line** of the Document.
  - It **specifies the version of the XML** specifications it is following.
  - It also **specifies the encodeing.**

    <?xml version="1.0" encoding="UTF-8"?>
  - The Character encoding allows us to specify the language based on the ISO standards or Unicode standards, which use to creat markup and contents of the documents.

Creating Well – Formed XML Documents

- **The root Element**
  - XML document must have exactly one root element.
  - Root element must be the first element immediately after the <?XML> tag

```
<?xml version="1.0" encoding="UTF-8"?>
<BOOKS>
    rest of the xml document
</BOOKS>
```

Creating Well – Formed XML Documents

- **Opening and Closing Tags (Element Tag Rules>**
  - All elements have an opening tag. Optionally, element also have a closing tag.

<BOOK Pubyear='1973'>

<BOOK_TITLE> LOOK Homeward </BOOK_TITLE>

<AUTHOR> Wolfe, Thomas </AUTHOR>

</BOOK>

Creating Well – Formed XML Documents

- **Empty Elements**
  - Empty Elements in XML can be represented in two ways.

(1) We can either use the tag pair <> and </> containing the element name to depict this, without content in between.

<Book_Name></Book_Name>

(2) We can just use the single tag </>

<Book_Name/>

Creating Well – Formed XML Documents

- **Empty Elements**
  - Show the customer name including the first and the last name, but the middle name should be empty.

<Name>

<First> Atul </First>

<Middle></Middle>

<Last> Patel </Last>

</Name>

Creating Well – Formed XML Documents

- **Opening and Closing Tags**
  - All elements have an opening tag. Optionally, element also have a closing tag.

<BOOK Pubyear='1973'>

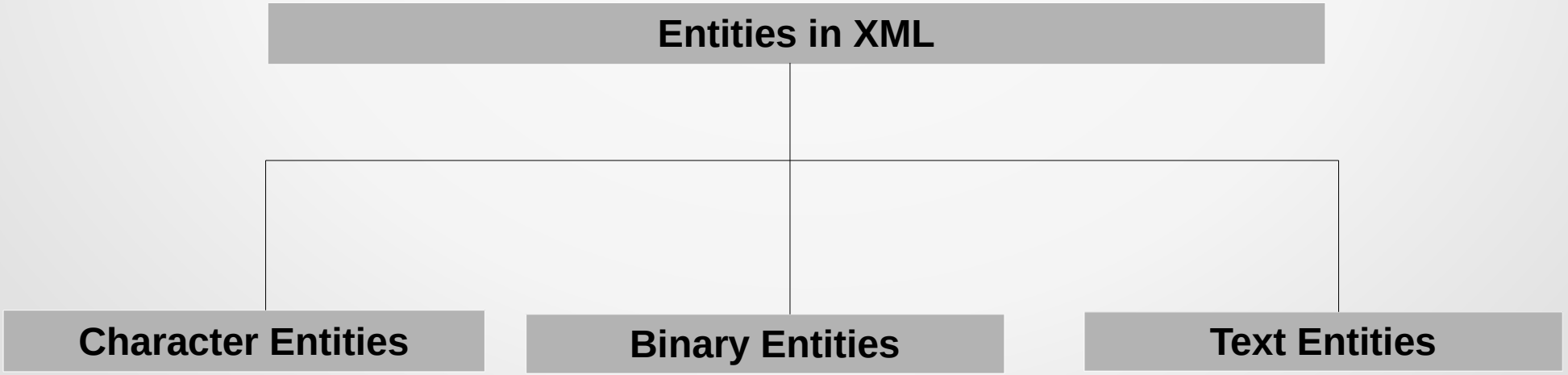
<BOOK_TITLE> LOOK Homeward </BOOK_TITLE>

<AUTHOR> Wolfe, Thomas </AUTHOR>

</BOOK>

Creating Well – Formed XML Documents

- **Entities**
  - An Entity inXML represents a text that you want to use repeatedly without having to write it every time.
  - We define it at one place, and refer to it from other place.
  - There are three types of entities in XML:



Creating Well – Formed XML Documents

- **Character Entities**
  - Character entity references are special character code that assign a different meaning to a special symbol.

Character Entity	Meaning
&amp;	& Character
&apos;	' Character
&gt;	> Character
&lt;	< Character
&quot;	“ Character

Creating Well – Formed XML Documents

- **Character Entities**
  - Character entity references are special character code that assign a different meaning to a special symbol.
  - **Contents in XML:**

    Please make sure that your offer is >$500
  - **Interpretation :**

    Please make sure that your offer is > $500

Creating Well – Formed XML Documents

- **Text Entities**
  - Text entities are used to associate large or repeated blocks of text with a name and replace the text with the entity name.
  - Declaring Syntax

    <!ENTITY name “content”>

    <!ENTITY country “INDIA”>

  - Demo4.xml

Creating Well – Formed XML Documents

- **Binary Entities**
  - Binary entities are used to associate a name with binary data (such as an image or a video) and use the entity name insted of the actual binary data.

<!ENTITY city SYSTEM “delhi.html” NDATA html>

Creating Well – Formed XML Documents

- **Element Nameing**
  - Should contain at least one latter : a-z or A-Z
  - Can start with an alphabet or an underscore
  - Can contain latters, digits, hypens, underscores, full stops
  - XML names are case sensitive.
  - Names cannot contain spces
  - Name cannot beused any prefix
  - i.e

    <Name05>

    <Name.05>

    <_05Name>

Creating Well – Formed XML Documents

- **Nesting Conventions**
  - In XML, child elements must be nested completely inside the parent element.
- **Adding Attributes**
  - Attributes allow us to specify more information about XML elements.
  - Attributes merely provide an alternative to sub-elements
  - Attributes conist of a name=”value” pair
  - Attributes are placed in the start tag of the element.
  - An element may have several attributes, each uniquely named.
  - Attributes must have a value
  - Values must be quoted with either double or single quotes

Creating Well – Formed XML Documents

- Comments

  <!-- THIS IS COMMENT IN XML -->

Element Content

- Element content is handled in one of two ways:

(1) Parsed Character Data (PCDATA): it is examined by the XML parser to discover XML content embedded within it.

Character Entity	Meaning
&amp;	& Character
&apos;	' Character
&gt;	> Character
&lt;	< Character
&quot	“ Character

Element Content

- Element content is handled in one of two ways:

  (2) Character Data (CDATA) : CDATA is not parsed and is treated as it is. It is useful for embedding other languages within the XML as :

  - HTML documents
  - XML documents
  - JavaScript documents
  - Etc.