0301402 INTRODUCTION TO XML

UNIT	MODULES	WEIGHTAGE
1	**Introduction to XML**	**20 %**
2	**Document Type Definition (DTD)**	**20 %**
3	**XML Namespace**	**20 %**
4	**XML Schema**	**20 %**
5	**Extensible StyleSheet Language (XSL)**	**20 %**

UNIT -2 XML DTD

- Introduction to DTD

- Why do we need DTDs?

- Types of DTD

- Inserting comments in a DTD

- Element Type Declaration

- Attribute Declaration

- Conditional Section

- Limitations of DTD

Introduction DTD

- **XML document that we contain bank account info**

<ACCOUNT>

    <ACC_NO> 0036895286*</ACC_NO>*

    <ACC_NAME> ATUL</ACC_NAME>

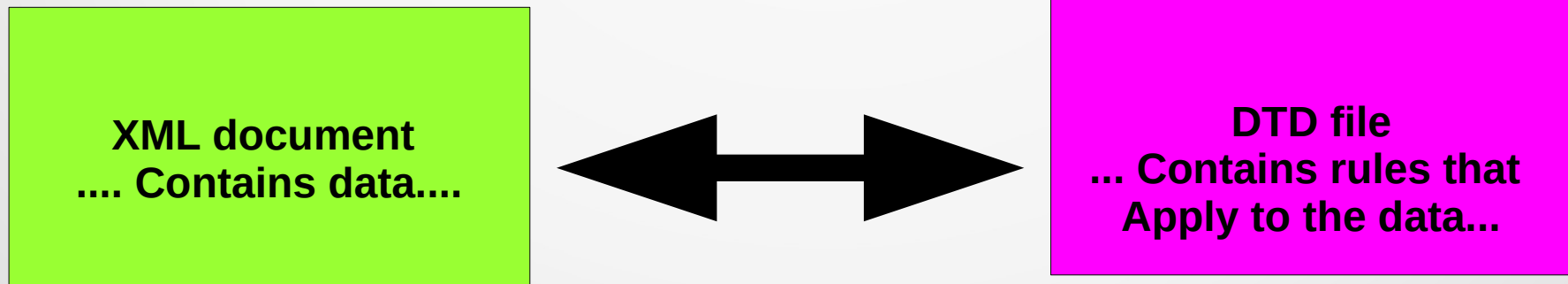   *<ACC_TYPE> SAVING </ACC_TYPE>*

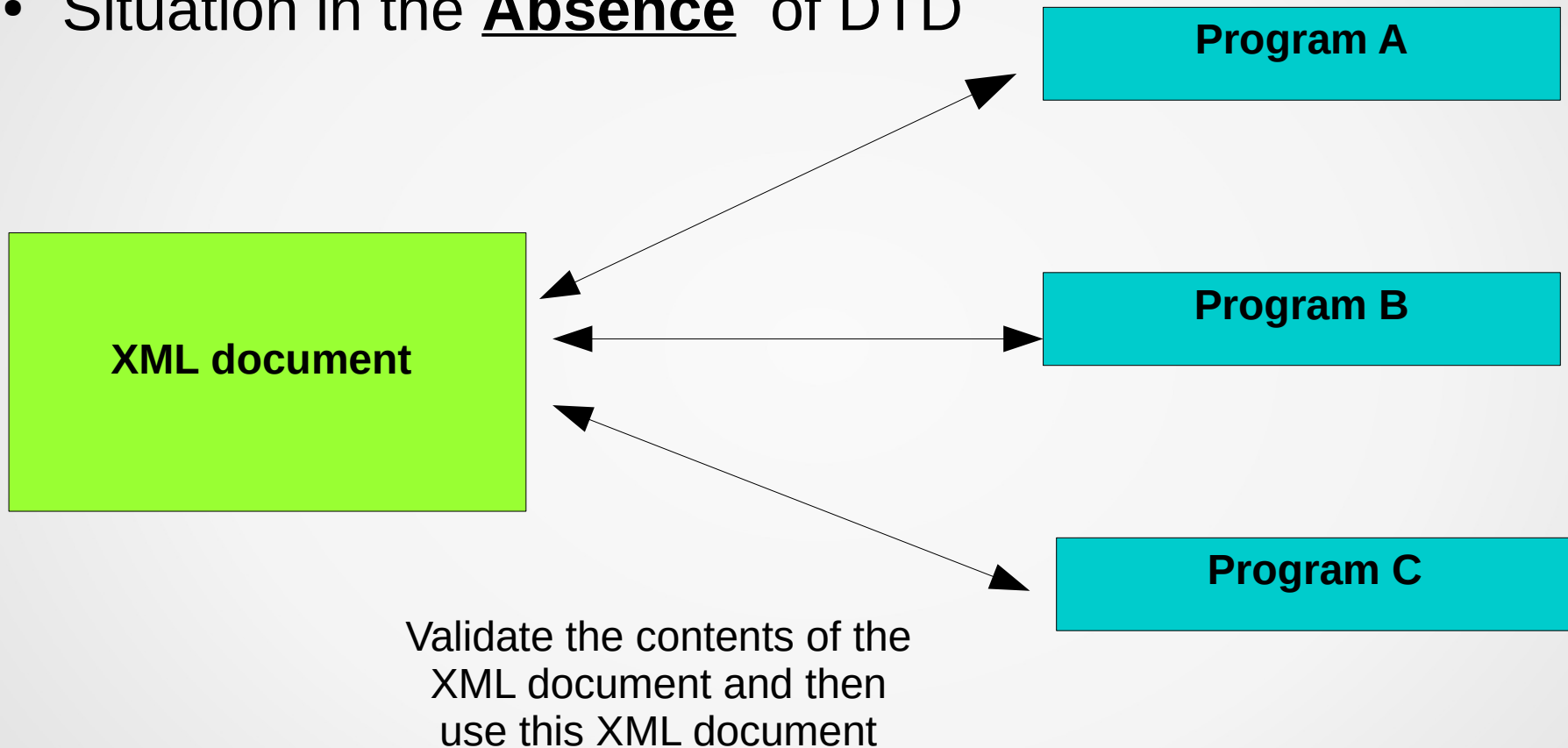   *<BOO_ID> T101Y06 </BOOK_ID)*

</ACCOUNT>

Introduction DTD

- DTD stand for Document Type Definitions

- A DTD allows us to **validate the contents of an XML** document.

- A DTD is usually a file with an **extension of .dtd**
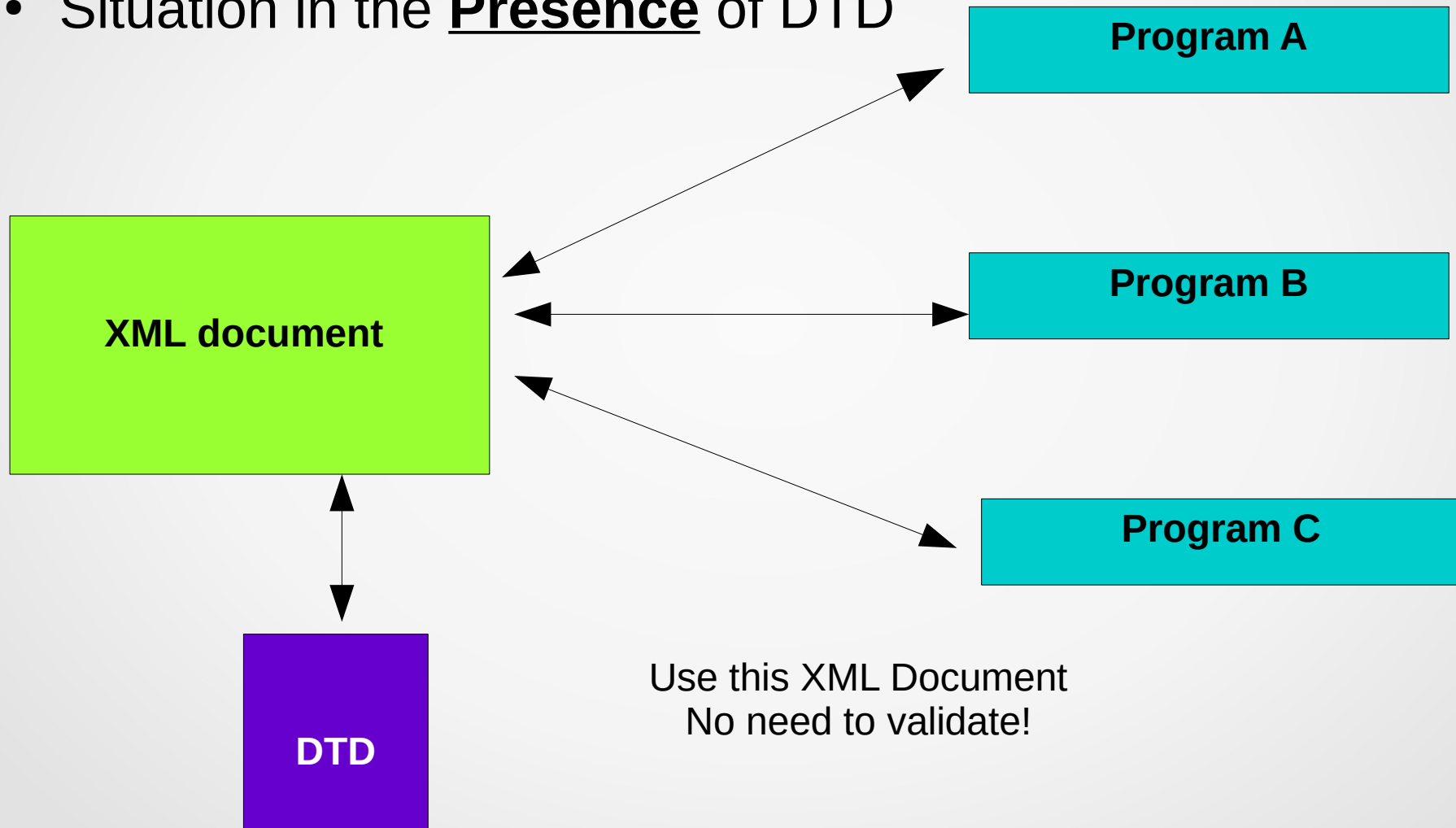
- It contains rules that apply to xml data.

XML document .... Contains data....

⬅➡

DTD file ... Contains rules that Apply to the data...

Introduction DTD

- Situation in the **Absence** of DTD

Program A

XML document

Program B

Program C

Validate the contents of the
XML document and then
use this XML document

Introduction DTD

- Situation in the **Presence** of DTD

Program A

XML document

Program B

Program C

DTD

Use this XML Document
No need to validate!

Need of DTD

- DTD **helps us in specifying the rules for validating the contents of an XML document at one place**, there by **allowing the application programs to concentrate on the processing of the XML** document.

- **Replace the use of resusable pieces** of text between 2 XML documents.

- **To verify the XML document.**

- **To meet the constraints.**

Types of DTD

- A **DOCTYPE** declaration in an XML document specifies that, xml file include a reference to a DTD file.

- *<!DOCTYPE rootelement "a.dtd">*

test.xml

a.dtd

<!DOCTYPE rootelement "a.dtd">

Rules related to the XML document (test.xml) for validation purposes

Types of DTD

- There are two types of DTDs:

  - Internal DTD

  - External DTD

    - System DTD

    - Public DTD

Types of DTD – Internal DTD

- An internal subset means the contents of the DTD are inside an XML document itself.

    <?xml version="1.0"?>

    <!DOCTYPE myBook [

    <!ELEMENT myBook (book_name) >

    <!ELEMENT book_name (#PCDATA) >

    ]>

    <myBook>

    <book_name> XML </book_name>

    </myBook>
    - **Demo1.xml**

Types of DTD – External DTD

- External DTD help us in two ways

  - It allow us to define a DTD once, and then refer to it from any number of XML Documents. Means Reusable.

  - It redue the size of the XML document.

Types of DTD – External DTD

- External DTD - xml document

<?xml version="1.0"?>

<!DOCTYPE myBook SYSTEM "mybook.dtd">

<myBook>

   <book_name> XML </book_name>

</myBook>

- **Demo2.xml**

Types of DTD – External DTD

- External DTD - dtd document

    <!ELEMENT myBook (book_name) >

    <!ELEMENT book_name (#PCDATA)>

Types of DTD – External DTD

- **System DTD (Private DTD):**

  – it has existance and relevance only in given text. Available in same computer or same network.

- **Public DTD**:

  – has usage beyond a single XML document. It is located on a different computer accessible via internet as a URL.

Element Type Declaration

- If we want to associate a DTD with an XML document, we need to declare all the elements that we would add in XML document.

  <!ELEMENT book_name (#PCDATA)>

- **book_name**: element name is generic identifier

- **#PCDATA** : data type content specification

Element Content Models

- Sequence, Occurrences, Choice

- Empty, Any, Mixed Content

Sequence

<!ELEMENT address(street, region, postal-code)>

<!ELEMENT street (#PCDATA)>

<!ELEMENT region (#PCDATA)>

<!ELEMENT postal-code (#PCDATA)>

- **Demo3.xml**

Choices

- **Specified using pipe(|) character**

  <!ELEMENT guest(name, beverage)>

  <!ELEMENT name (#PCDATA)>

  <!ELEMENT beverage tea|cofee>

  <!ELEMENT tea (#PCDATA) >

  <!ELEMENT coffee (#PCDATA) >

- **Demo4.xml**

Occurrences

- The number of occurrences or the frequency, of an element can be specified by either +, *, ?

- **+ : The element can occure one or more times**

- *** : The element can occure zero or more times**

- **? : The element can occure zero or one times**

- **Demo5.xml**

Empty

- An empty element is the one that can neither contain any data nor any sub-elements.

- Empty elements

- <!ELEMENT name EMPTY>

    <name></name>

    </name>

Mixed Content

- An element can contain either some text or other sub element.

- Mixed content elements

- <!ELEMENT name (sub element | #PCDATA)>

  <mybook>possessing book is very rich!

  <book>Operating systems</book>

  </mybook>

- **Demo6.xml**

ANY Content

- An element declared with ANY type can contain any content, including PCDATA, sub-elements, combinations of the two or empty element.

- **<!ELEMENT mybook ANY>**

- *<mybook>XML related tech*

    <book></book>

    <book>XML 1 </book>

</mybook>

- ***Demo7.xml***

Attribute Declaration

- Elements describe the markup of an XML document.

- Attribute provide more details about the elements.

- An element can have zero or more attributes.

- **The keyword "ATTLIST" describes the attributes(s) for an element.**

- **Demo_20.xml**

Attribute Declaration – Default Attributes

- There are three type of default values that can specified to attributes.

  - #IMPLIED

  - #REQUIRED

  - #FIXED

- #IMPLIED – specifies that if the **attribute does note appear** in the element, the a**pplication using the XML document is free to decide the value** for the attribute.

Attribute Declaration – Default Attributes

- #REQUIRED – specifies that the **attribute must appear inside the element.**

- #FIXED – specifies that the **attribute value is costant.** That is, the attribute value as defined in the DTD must be exactly the same in the corresponding XML document.

-

- DEMO_21.xml

Types of Attributes

- Attributes can be three types
  - String
  - Tokenised
    - ID
    - IDREF
    - ENTITY
    - NMTOKEN
  - Enumerated

Types of Attributes

- **String Type Attribute**

  - String attributes are declared by using the CDATA (Character Data) keyword.

  - **An string attribute can contain any values except < > & ' "**

Types of Attributes

- **Tokensied Type Attribute**

  - **Tokenised attributes specify certian restictions that get applied to the attribute values.**

  - **ID: it uniquely identifies an element.** This is similar to the concept of a primary key.

    ID type attribute in DTD = Primary Key column in tabe

Types of Attributes

- **Tokensied Type Attribute**

  - **IDREF: it refers to elements with a specific ID attribute value.** This is similar to the concept of a foreign key.

    IDREF type attribute in DTD = Foreign Key column in tabe

  - *When we use **IDREFS**, insted of IDREF, it indicates that for one ID, we want to have multiple references.*

  - **DEMO22.xml**

Types of Attributes

- **DEMO22.xml**

Types of Attributes

- **Tokensied Type Attribute**
  - **ENTITY :** it useful in helping us define reusable pieces of text.
  - **DEMO23.xml**

Types of Attributes

- **Tokensied Type Attribute**

  – **NMTOKEN :** An NMTOKEN attribute mandates certain restrictions to be followed.

    - NMTOKEN stands for "name token"
    - **It could contian letters, digits, periods, underscores, hyphens, colon.**

    <!ATTLIST office phone NMTOKEN #REQUIRED>

          <office phone="91-79-268597"></office>

    - **DEMO23.xml**

Types of Attributes

- **Enumerated Type Attribute**

  - An enumerated **attribue allows us to specify the list or set of possible values** that the attribute can have.

  - DEMO_24.xml

  - DEMO_25.xml

Conditional Sections

- The **condition section come into picture only if the specified condition is satisfied,** otherwise, the original DTD is in effect.

<![keyword

      [conditional DTD declarations]

  ]>

Conditional Sections

- ***INCLUDE key word***

  - When we specify INCLUDE, the conditional declarations inside this section are considered for validations.

- ***IGNORE Key word***

  - When we specify IGNORE, the conditional declarations inside this section are not considered for validations.

Limitations of DTDs

- *Non XML syntax*

- One DTD per XML

- Weak data typing

- No inheritance

- Overridding a DTD : internal can override external DTD

- No DOM support