Document Type Definition (DTD)

What is DTD?

- **DTD is used to declare each of the building blocks (elements) used in a XML document**

- **DTD defines:**
  - **a structure of the XML document**
  - **a list of legal elements of the XML document**

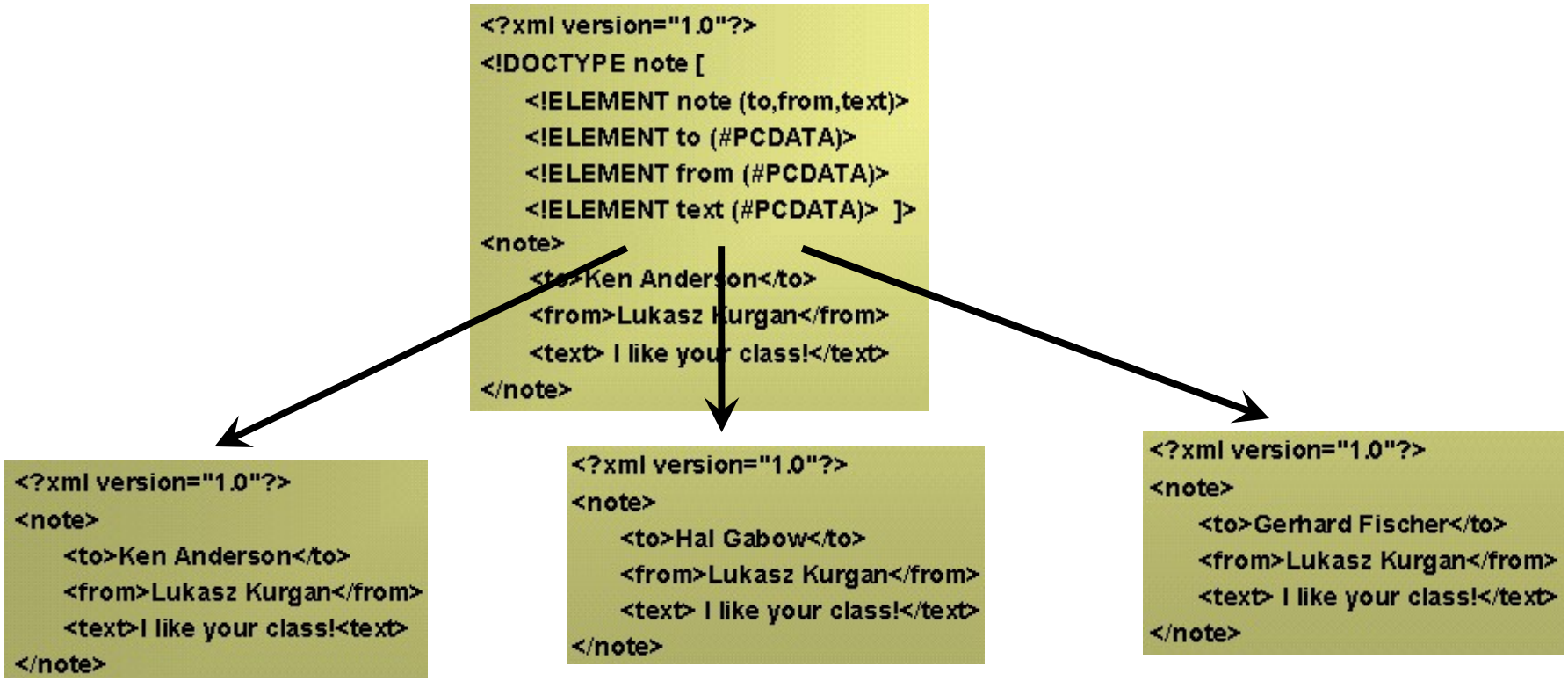
Well-Formed vs. Valid Document

- **Well-formed document – the document that adheres to the XML syntax rules**

- **Valid document – the document that adheres to the rules defined in the corresponding DTD document**

Only the valid documents are valuable in terms of sharing and retrieving information.

Internal vs. External DTD



Internal vs. External DTD

- **External DTD are better because of:**
  - **possibility of sharing definitions between XML documents**
  - **The documents that share the same DTD are more uniform and easier to retrieve**
- **Linking in the DTD document**

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM “note.dtd">
<note>
    <to>Ken Anderson</to>
    <from>Lukasz Kurgan</from>
    <text>Ok! We can see some progress</text>
</note>
```

Building blocks of XML

- **XML documents (and HTML documents) are made up by the following building blocks:**

	Description	Examples in HTML	Examples in XML
Elements	**main building blocks**	***body,table***	***note, from***
Tags	**used to markup elements**	***<body></body>***	***<from></from>***
Attributes	**provide extra information about elements**	******	***<note att="abc.xml" />***
Entities	**variables used to define common text**	*** & " '***	
PCDATA	**PCDATA means parsed character data. PCDATA is text that will be parsed by a parser. Tags inside the text will be treated as markup and entities will be expanded.**		
CDATA	**CDATA means character data. CDATA is text that will NOT be parsed by a parser. Tags inside the text will NOT be treated as markup and entities will not be expanded.**		

Declaring Elements in DTD

An element declaration has the following syntax:

<!ELEMENT element-name (element-content)>

- **Elements name cannot include <> characters and must start with letter or underscore**
- **Elements name can include namespace declaration: Namespace:element-name**

Declaring Elements in DTD

Details on element declarations:

Empty elements

```
<!ELEMENT element-name (EMPTY)>
```

Elements with data

```
<!ELEMENT element-name (#CDATA)>
<!ELEMENT element-name (#PCDATA)>
<!ELEMENT element-name (ANY)>
```

Elements with children (sequences)

```
<!ELEMENT note (to,from,text)>
<!ELEMENT to (#CDATA)>
<!ELEMENT from (#CDATA)>
<!ELEMENT text (#CDATA)>
```

Declaring Elements in DTD

Details on element declarations:

Declaring zero or more occurrences of the same element

<!ELEMENT element-name (child-name*)>

Declaring minimum one occurrence of the same element

<!ELEMENT element-name (child-name+)>

Declaring mixed content

<!ELEMENT note (to+,from,message*,#PCDATA)>

The example above declares that the element note must contain at least one *to* child element, exactly one *header*, zero or more *message*, and some other parsed character data.

Declaring Attributes in DTD

XML element attributes are declared with an ATTLIST declaration. An attribute declaration has the following syntax:

<!ATTLIST element-name attribute-name attribute-type default-value>

	value	explanation
attribute-type	CDATA	character data
	(eval|eval|..)	enumerated value
	ID	unique id
	NMTOKEN	valid XML name
default-value	#DEFAULT value	default value
	#REQUIRED	must be included in the element
	#IMPLIED	does not have to be included
	#FIXED value	value is fixed

Declaring Attributes in DTD

Attribute declaration example

DTD example:

```
<!ELEMENT square EMPTY>
<!ATTLIST square width CDATA "0">
```

XML example:

```
<square width="100"></square>
```

Entities in DTD

- **Variables used to define shortcuts to common text**
- **Entity references are references to entities**
- **Can be declared internally or externally**

Internal Entities in DTD

Define shortcuts to common text

Syntax:

```
<!ENTITY entity-name "entity-value">
```

Example:

In DTD:

```
<!ENTITY writer “Robert Eckstein">
<!ENTITY copyright “&#xA9;”>
```

In XML:

```
<author>&copyright; &writer;</author>
```

External Entities in DTD

Allow to copy the XML content located at specified URI into the current XML document

Syntax:

```
<!ENTITY entity-name SYSTEM "URI/URL">
```

Example:

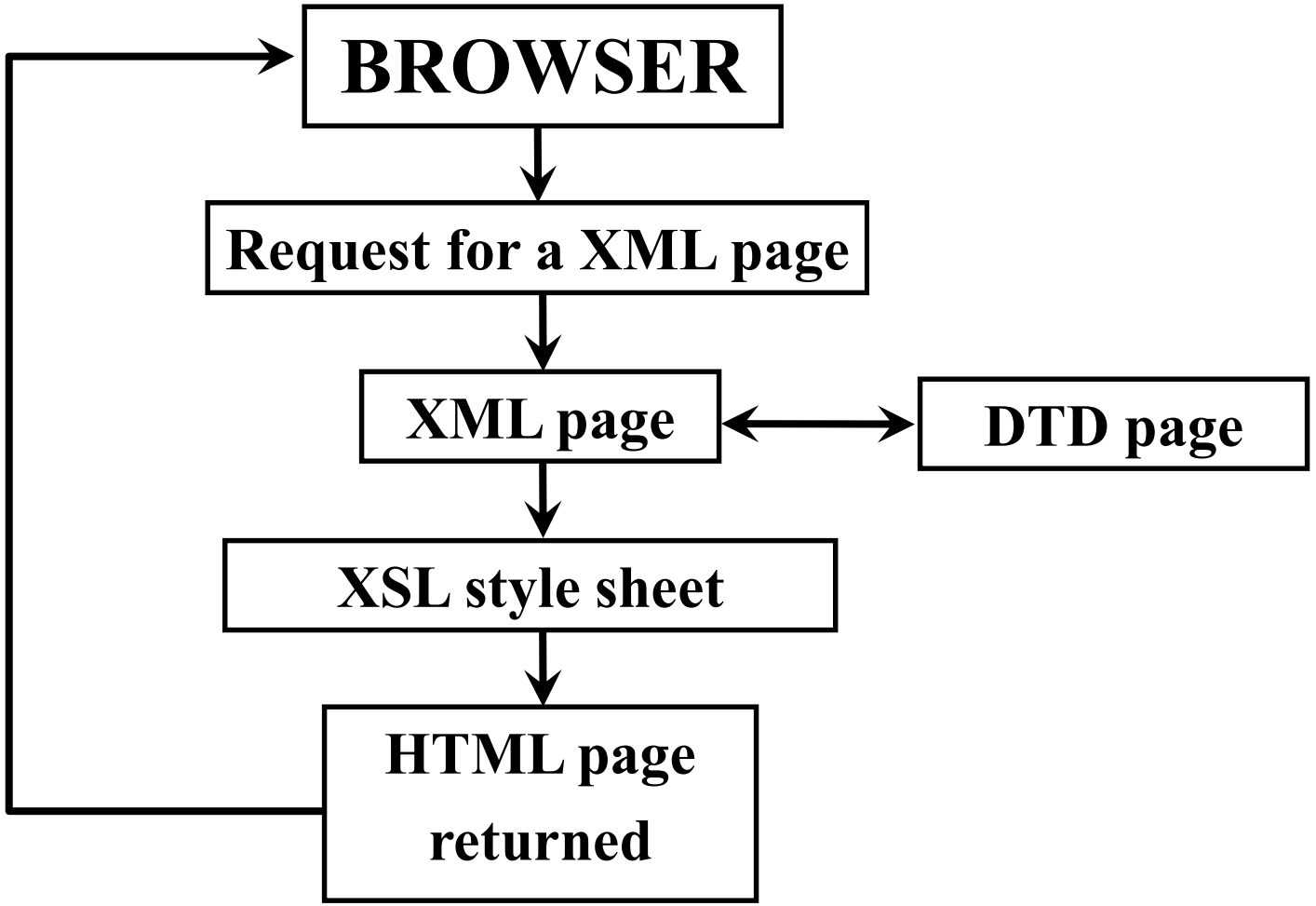
In DTD:

```
<!ENTITY article SYSTEM “http://www.articles.com/DTD.xml">
```

In XML:

```
<articles_xml>
 <heading>Article from www.articles.com</heading>
 &article;
</articles_xml>
```

HTML=XML+DTD+XSL



“Bigger” Example

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE MLBibliographies SYSTEM "default.dtd">
<MLBibliographies>
    <PageTitle>Machine Learning Bibliographies</PageTitle>
    <PageSubTitle>Maintained by Lukasz Kurgan</PageSubTitle>
    <Category href="FeatureSelection.xml"> 1. Feature Selection</Category>
    <Category href="RuleInduction.xml"> 2. Rule Induction</Category>
    <Category href=""> 3. Discretization</Category>
    <Category href=""> 4. Learning Ensemble of Classifiers</Category>
    <LastUpdate> 09/11/01</LastUpdate>
</MLBibliographies>
```

DTD document

```
<!ELEMENT MLBibliographies (PageTitle, PageSubTitle, Category*, LastUpdate, Publication*)>
<!ELEMENT PageTitle (#PCDATA)>
<!ELEMENT PageSubTitle (#PCDATA)>
<!ELEMENT Category (#PCDATA)>
<!ELEMENT LastUpdate (#PCDATA)>
<!ATTLIST Category href CDATA #IMPLIED>
```

Why use the DTD?

- XML provides an application independent way of sharing data
- With a DTD, independent groups of people can agree to use a common DTD for interchanging data
- Your application can use a standard DTD to verify that data that you receive from the outside world is valid
- You can also use a DTD to verify your own data