Module 2

Namespaces

Module Overview

In this module, you will learn about:

- XML Namespaces
- Working with Namespaces Syntax

Lesson 1 – XML Namespaces

In this first lesson, **XML Namespaces**, you will learn to:

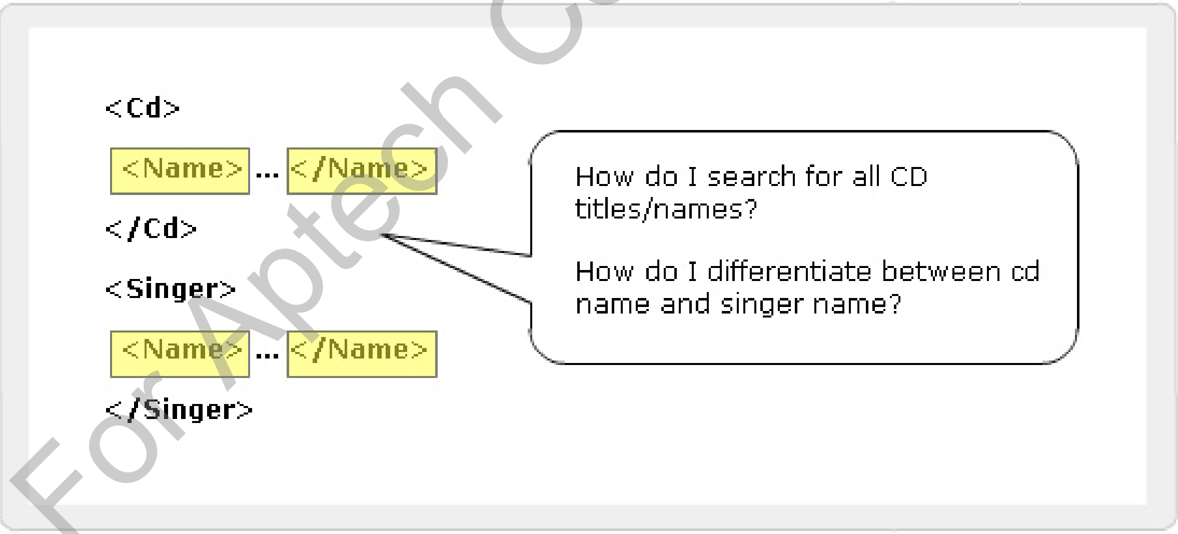
- Identify the need for a namespace.
- Define and describe namespaces in XML.

Duplicate Element Names

- It allows developers to create their own elements and attributes for their own projects.
- Developer has to ensure the uniqueness of the element names and attributes in a document.

Consequences of Duplicate Element Names

- Name conflicts are inevitable from different developers.
- It is difficult for the browser to distinguish a conflicting element.



```
<Cd>
<Name> ... </Name>
</Cd>
<Singer>
<Name> ... </Name>
</Singer>
```

How do I search for all CD titles/names?

How do I differentiate between cd name and singer name?

Namespaces

- Elements are distinguished by using namespaces.
- A namespace is a collection of names.
- Namespaces allow the browser to:
  - Combine documents from different sources
  - Identify the source of elements or attributes

Lesson 2 – Working with Namespaces Syntax

In this last lesson, **Working with Namespaces syntax**, you will learn to:

- Explain the syntax for XML namespaces.
- Discuss attributes and namespaces.
- Discuss how to use default namespaces.

Prefixing element names

- Prefixes in element names provide a means to prevent name collisions.

Code Snippet

```
<CD:Title> Feel </CD:Title>
```

and

```
<Book:Title> Returning to Earth </Book:Title>.
```

In the above example, both `CD` and `Book` are namespace prefixes.

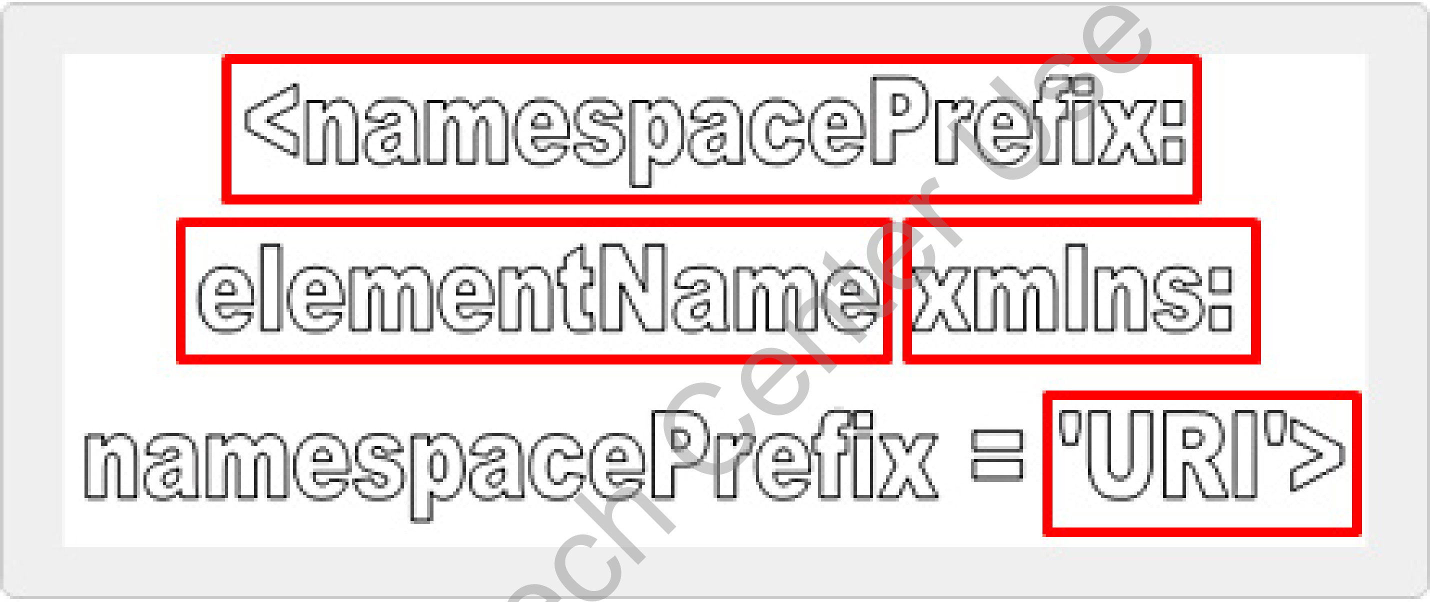
Problems Posed by Prefixes

- Duplication would still exist if prefixes are not unique
- To solve this problem, each namespace prefix is added to a Uniform Resource Identifier (URI)

Code Snippet

```
<S:Student xmlns:S=” http://www.spectrafocus.com
/student/”>
  <S:First>John</S:First>
  <S:Last>Dewey</S:Last>
  <S:Title>Student</S:Title>
</S:Student>
```

Namespace Syntax 1-3



<namespacePrefix:
elementName xmlns:
namespacePrefix = 'URI'>

Namespace Syntax 2-3

NamespacePrefix

- Used as a reference to the namespace
- Prefixes must not begin with `xmlns` or `xml`

ElementName

- Specifies the name of the element

xmlns

- `xmlns` stands for XML namespace

URI

- URI is a string of characters which identifies an Internet Resource

Namespace Syntax 3-3

Code Snippet

```
<Auc:Books
xmlns:Auc=”http://www.auction.com/books”
xmlns:B=”http://www.books.com/HTML/1998/xml1”>
...
<Auc:BookReview>
<B:Table>
...
```

Placing attributes in a Namespace 1-3

- Attributes belong to particular elements
- They are not a part of namespace, even if the element is within some namespace
- If an attribute has no prefix, it has no namespace
- An attribute without a prefix is in default namespace
- If an attribute name has a prefix, its name is in the namespace indicated by the prefix

Placing attributes in a Namespace 2-3

Syntax

```
<Catalog xmlns:Book =
“http://www.aptechworldwide.com”>
<Book:Booklist>
<Book:Title Book:Type = “Fiction”>Evening in
Paris</Book:Title>
<Book:Price>$123</Book:Price>
</Book:Booklist>
</Catalog>
```

Placing attributes in a Namespace 3-3

Code Snippet

```
prefix:localname=’value’
or
prefix:localname=”value”
```

where,

- `prefix` is used as a reference to the namespace. Prefixes must not begin with `xmlns` or `xml`
- `localname` is the name of an attribute
- `value` mentions a user defined value for an attribute

Default Namespaces 1-2

MathML Document

- XML-based markup language to represent complex mathematical expressions
- Comes in two types:
  - As a markup language for presenting the layout of mathematical expressions
  - As a markup language for presenting the mathematical content of the formula

Code Snippet

```
<MRow>
  <Mi>x</Mi>
  <Mo>+</Mo>
  <Mn>1</Mn>
</MRow>
```

Default Namespaces 2-2

Syntax

```
<elementName xmlns='URL'>
```

where,

`elementName` specifies the name of the element belonging to the same namespace

`URL` specifies the namespace which is reference for a document or an HTML page on the Web

Code Snippet

```
<Catalog xmlns:Book = "http://www.aptechworldwide.com">
<Book:Booklist>
   <Book:Title Book:Type = "Fiction">Evening in
   Paris</Book:Title>
   <Book:Price>$123</Book:Price>
   </Book:Booklist>
</Catalog>
```

A default namespace using the `xmlns` attribute with a URI as its value

Override Default Namespaces 1-2

- Default namespace applies to the element on which it was defined and all descendants of that element
- New namespace definition overrides the previous one and becomes the default for that element

Override Default Namespaces 2-2

Code Snippet

```
<Catalog xmlns = “http://www.aptechworldwide.com”>
 <Book>
  <Title type = “Fiction”>Evening in Paris</Title>
  <Price>$123</Price>
 </Book>
 <Book>
  <Title type = “Non-Fiction”>Return to Earth</Title>
  <Price xmlns = “http://www.aptech.ac.in”>$23</Price>
  <Title type = “Non-Fiction”>Journey to the center of
the Moon</Title>
  <Price>$123</Price>
 </Book>
</Catalog>
```

This namespace of `price` element applies only to it and overrides the namespace in the `catalog` element

Summary

- **XML Namespaces**
  - Namespaces distinguish between elements and attributes with the same name from different XML applications.
  - It is a collection of names that can be used as element names or attribute names in XML document.
  - XML namespaces provide a globally unique name for a element or attribute to avoid name collisions.
- **Working with Namespaces syntax**
  - Namespaces are declared by an `xmlns` attribute whose value is the URI of the namespace.
  - If an attribute name has no prefix, it has no namespace.
  - A default namespace is used by an element and its child elements if the element does not have a namespace prefix.