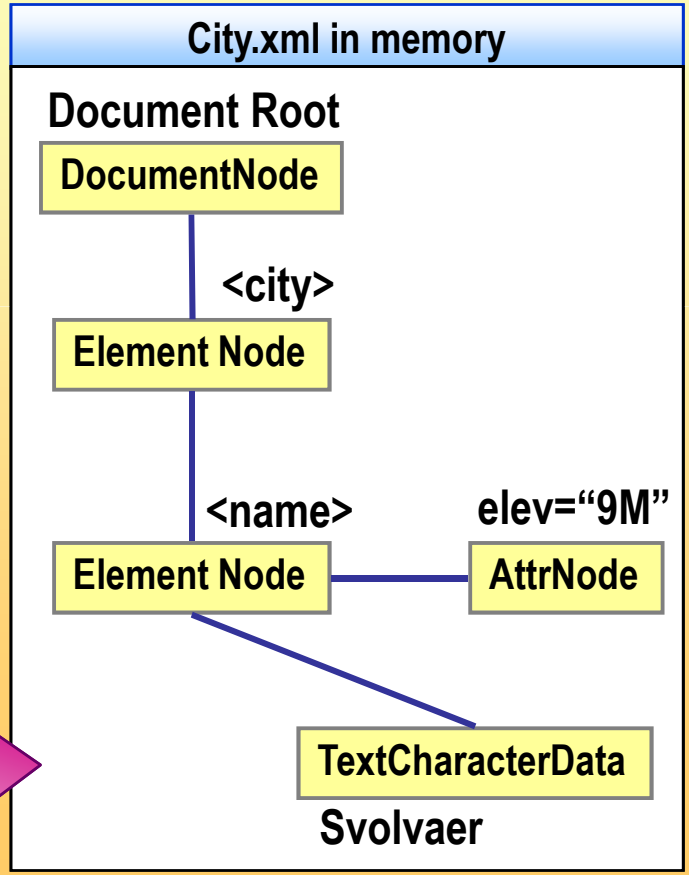
◆XML DOM(Document Object Model)소개

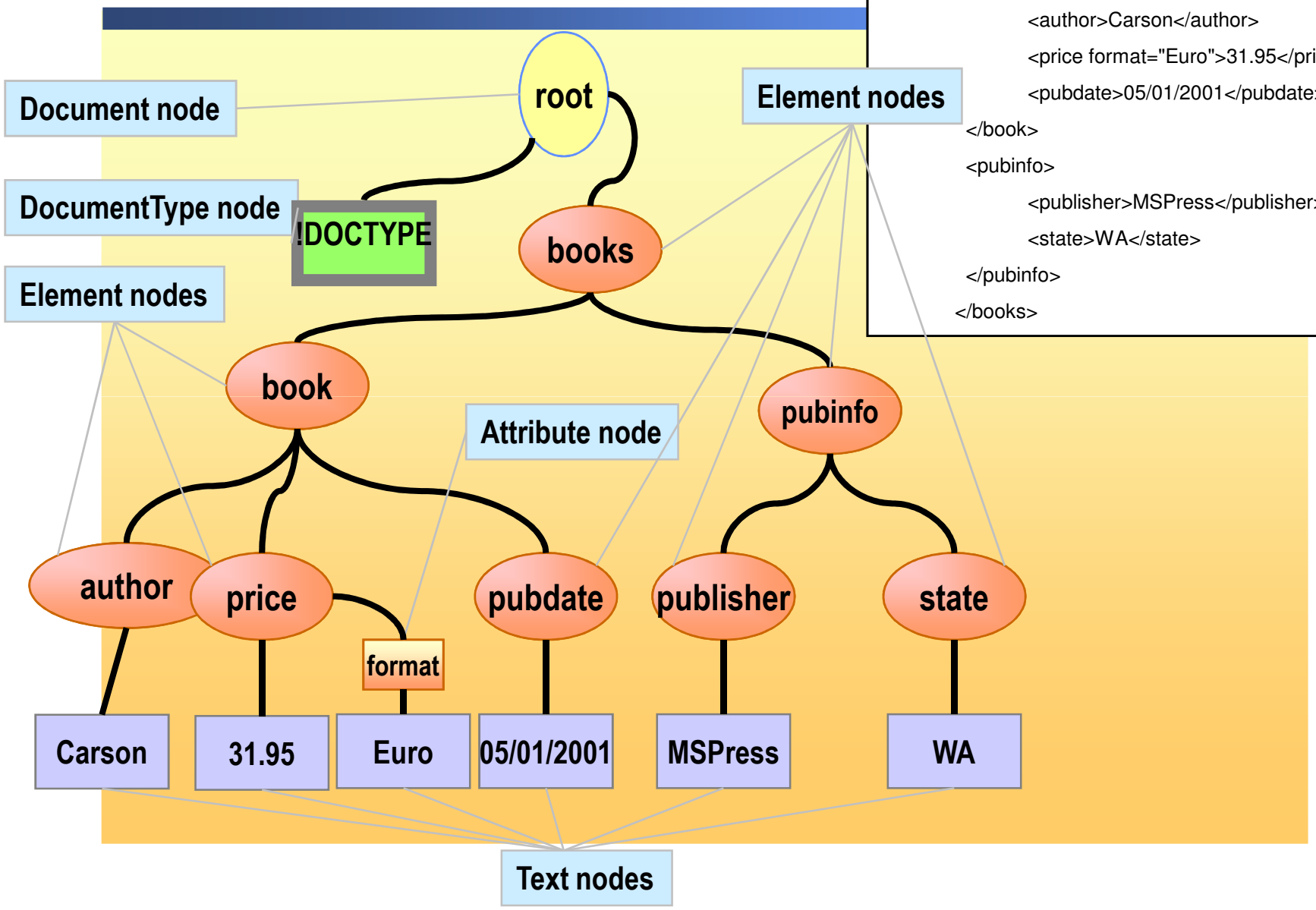
- DOM is the W3C programming interface for XML
- DOM models an XML source as a tree of nodes in memory
- You can use DOM to:
  - Navigate and search
  - Add and delete content

City.xml

```
<city>
<name lev="9M">Svolvaer</name>
</city>
```



DOM Nodes Correspond to XML



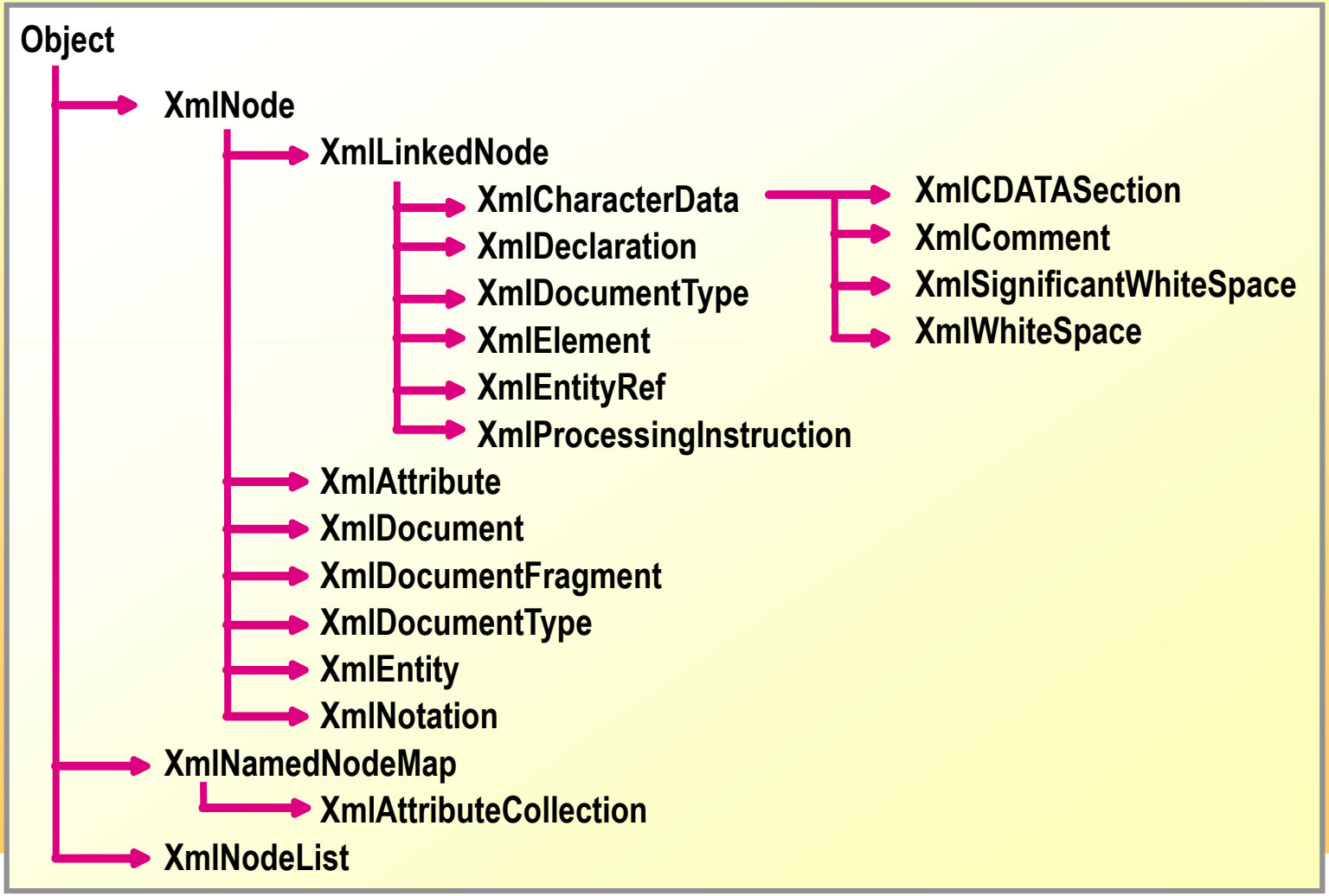

```
<?xml version="1.0"?>
<books>
  <book>
        <author>Carson</author>
        <price format="Euro">31.95</price>
        <pubdate>05/01/2001</pubdate>
  </book>
  <pubinfo>
        <publisher>MSPress</publisher>
        <state>WA</state>
  </pubinfo>
</books>
```

DOM Nodes and Related .NET Node Types

W3C DOM node types	Related .NET DOM node type
Document	XmlDocument
DocumentFragment	XmlDocumentFragment
DocumentType	XmlDocumentType
EntityReference	XmlEntityReference
Element	XmlElement
Attr	XmlAttribute
ProcessingInstruction	XmlProcessingInstruction
Comment	XmlComment
Text	XmlText
CDATASection	XmlCDATASection
Entity	XmlEntity
Notation	XmlNotation

What .NET Classes Support the DOM?

.NET Framework DOM class hierarchy

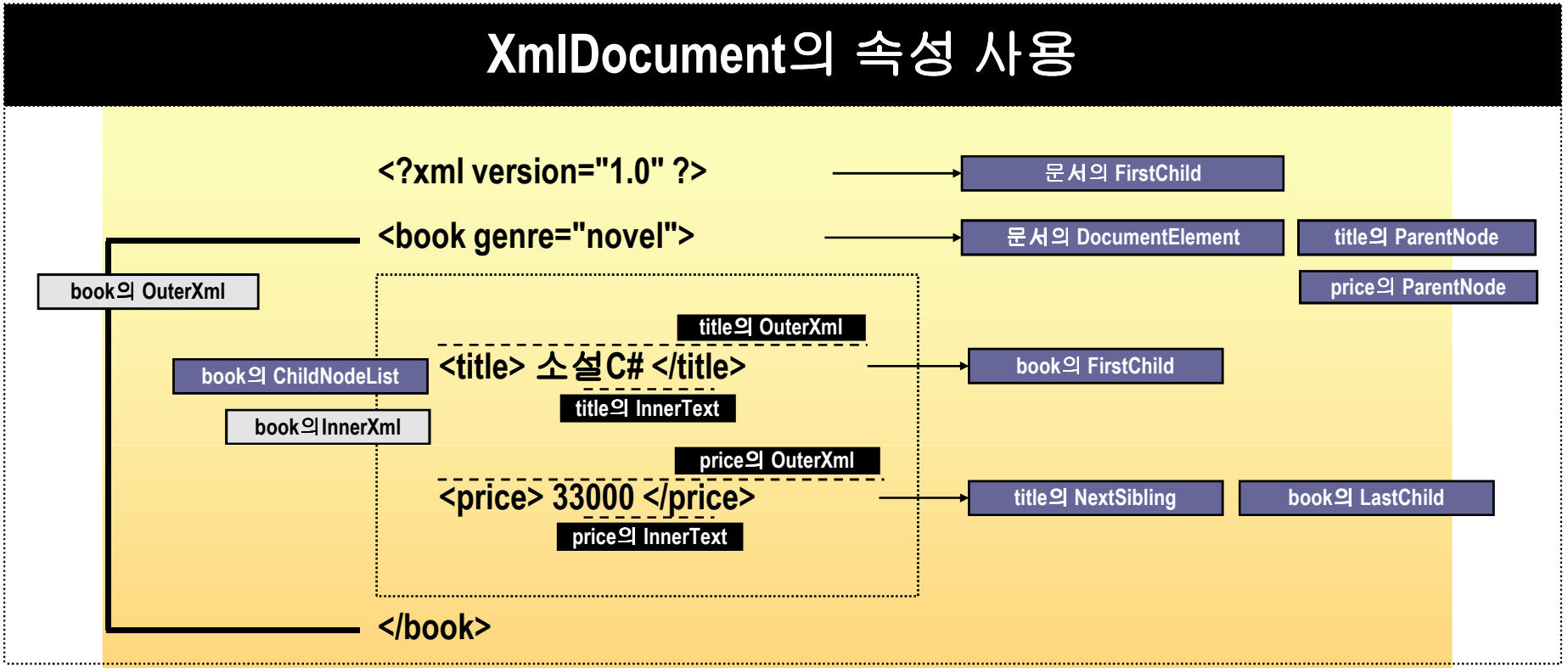


◆XML문서를 DOM으로 가져오기

- XmlDocument 를 사용

To load XML from this type:	Use this method:
String	LoadXml()
Stream File XmlReader TextReader	Load()

XmlDocument로 XML 읽어오기 및 노드 검색




```
XmlDocument doc = new XmlDocument();
doc .Load("c:\\xml\\books.xml"); //XML 파일 로딩
Console.WriteLine(doc.FirstChild.OuterXml); //<?xml version="1.0"?>
Console.WriteLine(doc.DocumentElement.Name); //book
Console.WriteLine(doc.DocumentElement.OuterXml);  //<book genre="novel"><title>소설C#</title><price>33000</price></book>
Console.WriteLine(doc.DocumentElement.InnerXml); //<title>소설C#</title><price>33000</price>
XmlNode pi = doc.FirstChild;
XmlNode book = doc.DocumentElement;
XmlNode title = book.FirstChild;
```

XmlDocument의 재귀적 노드 검색

- **XmlDocument 생성**

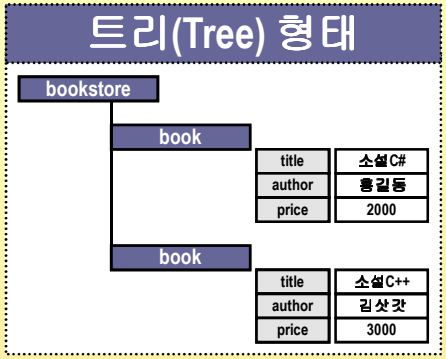
```
XmlTextReader tr = new XmlTextReader ("sample.xml");
tr.WhitespaceHandling = WhitespaceHandling.None;
XmlDocument xdoc = new XmlDocument();
xdoc.Load (tr);
```

- **재귀 함수 호출**

```
DisplayTree(xdoc.DocumentElement);
```

- **재귀함수**

```
public static void DisplayTree(XmlNode node){
    if (!node.HasChildNodes)
        Console.WriteLine(node.Value);
    else
            Console.WriteLine("<"+ node.Name+">");
    if (node.HasChildNodes){
        node = node.FirstChild;
        while (node != null){
                DisplayTree(node);
                node = node.NextSibling;
        }
    }
}
```



◆XmlDocument를 이용한 노드검색

- **XmlDocument의 간단한 노드 검색 함수**
  - SelectNode()
  - SelectSingleNode()
- **XmlNodeList SelectNode(XPath쿼리)**
  - XPath 쿼리에 해당하는 XmlNode 목록을 XmlNodeList 형태로 리턴한다.
- **XmlNode SelectSingleNode(XPath쿼리)**
  - XPath 쿼리에 해당하는 첫번째 XmlNode를 리턴한다.
  - [참고] XPath 쿼리는 XML을 검색하기 위한 쿼리
- **/bookstore/book/@ISBN**
  - bookstore 하위에 존재하는 book의 ISBN Attribute
  - @는 Attribute를 의미한다.

```
XmlNodeList nodeList = root.SelectNodes("/bookstore/book/@ISBN");
foreach (XmlNode isbn in nodeList)
        MessageBox.Show(isbn.Value);
```

- **descendant::book[@publicationdate='1967']**
  - 현재의 엘리먼트가 존재하는 하위의 book 엘리먼트의 속성 중 publicationdate가 1967인 노드

```
XmlNode book;
book = root.SelectSingleNode("descendant::book[@publicationdate='1967']");
```

XmlDocument를 이용한 노드검색

```
XmlNodeList cheapBooks = doc.SelectNodes("//book[price<20000]");
foreach (XmlNode b in cheapBooks)
        Console.WriteLine(b.InnerText );
```

- **GetElementsByTagName()**
  - Element를 Name으로 얻기

```
XmlNodeList n = doc.GetElementsByTagName("book");
foreach (XmlNode b in n)
        Console.WriteLine(b.InnerXml  );
```

◆XmlDocument로 XML 수정하기

- Setting the .InnerText property

```
XmlNode root = doc.DocumentElement;
XmlElement book;
book =
    (XmlElement)root.SelectSingleNode("/book/title");
book.InnerText = "Emma";
```

- The XmlNode.ReplaceChild() method
- The XmlElement.SetAttribute() method

◆XmlDocument로 노드 삭제하기

- To remove an element, use XmlNode.RemoveChild()

```
XmlNode firstBook =
doc.SelectSingleNode("//book");
firstBook.ParentNode.RemoveChild(firstBook);
```

- To remove an attribute, use XmlElement.RemoveAttribute()

```
XmlNode book = doc.SelectSingleNode("//book");
book.RemoveAttribute("isbn");
```

- To delete all child nodes, use XmlNode.RemoveAll()

◆XmlDocument로 저장 - Element Node 생성하기

- Use XmlDocument.CreateElement()
- For simple elements, create text content first

```
XmlDocument doc = new XmlDocument();
XmlNode bookNode = doc.CreateElement("book");

XmlNode titleNode = doc.CreateElement("title");
XmlNode title = doc.CreateTextNode("XML is Cool");
titleNode.AppendChild(title);

XmlNode priceNode = doc.CreateElement("price");
XmlNode price = doc.CreateTextNode("19.99");
priceNode.AppendChild(price);

bookNode.AppendChild(titleNode);
bookNode.AppendChild(priceNode);
```

Element Node에 Attributes 지정

- To set attributes for an Element node, use:

  XmlElement.SetAttribute()

```
bookNode.SetAttribute("sku", "XYZ-123");
```

- CreateAttribute and SetAttributeNode methods

다른 종류의 Nodes 생성하기

- Comments

```
XmlComment com = doc.CreateComment("Woo Hoo!");
```

- CDATA sections

```
string title = "Bill & Ted's <XML> Adventure";
XmlCDataSection cd = oc.CreateCDataSection(title);
```

- Processing instructions

```
string name = "xml-stylesheet";
string target = "type='text/xsl` href='style.xsl'";
XmlProcessingInstruction xpi = null;
xpi = doc.CreateProcessingInstruction(name, target);
```

XmlDocument로 XML 저장하기 예제

book

```
XmlElement book = xdoc.CreateElement("book");
book.SetAttribute("genre", "Mystery");
book.SetAttribute("publicationdate", "2001");
book.SetAttribute("ISBN", "56-6456-64");
```

```
<book genre="Mystery" publicationdate="2001" ISBN="56-6456-64" xmlns="">
</book>
```

전체 연결하기

1. book.AppendChild(title);
2. book.AppendChild(author);
3. book.AppendChild(price);
4. author.AppendChild(fname);
5. author.AppendChild(lname);
6. xdoc.DocumentElement.Append(book);

```
XmlTextWriter tw = new XmlTextWriter("mytest.xml", null);
tw.Formatting = Formatting.Indented;
xdoc.WriteContentTo(tw);
tw.Close();
```

title

```
XmlElement title = xdoc.CreateElement("title");
title.InnerText = "Novel C#";
```

```
<title>Novel C#</title>
```

author

```
XmlElement author = xdoc.CreateElement("author");
```

```
<author></author>
```

price

```
XmlElement price = xdoc.CreateElement("price");
price.InnerText = "300000";
```

```
<price>300000</price>
```

first-name

```
XmlElement fname = xdoc.CreateElement("first-name");
fname.InnerText = "Hong ";
```

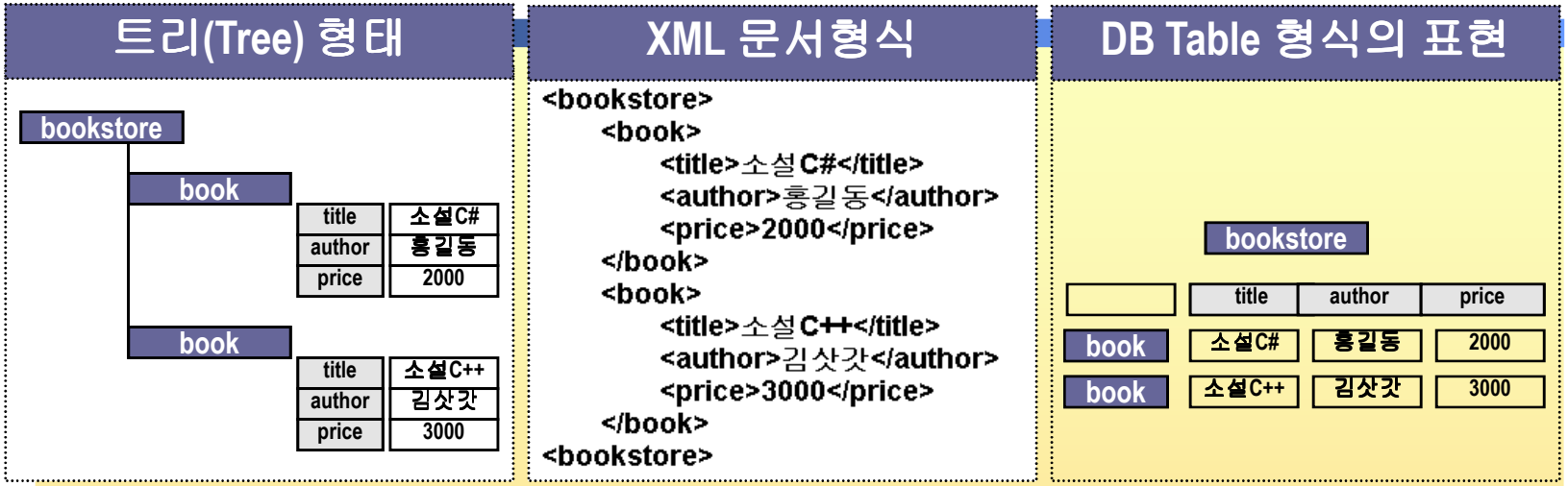
```
<first-name>Hong </first-name>
```

last-name

```
XmlElement lname = xdoc.CreateElement("last-name");
lname.InnerText = "Gil Dong";
```

```
<last-name>Gil Dong</last-name>
```

◆ XPath란?



- **XPath**
  - XML 데이터를 검색하기 위한 쿼리를 표현하는 방법
  - Search, Filter, Summarize
- **XPathExpression 클래스**
  - XPath를 구현한 클래스
- **XPathNavigator 클래스**
  - XPath를 만들거나 처리하기 위한 클래스

XPath 관련 클래스들

```
public interface IXPathNavigable{
        XPathNavigator CreateNavigator();
}
```

- **IXPathNavigable**
  - IXPathNavigable 내의 CreateNavigator() 함수는 XPathNavigator를 리턴
- **XPathNavigator**
  - Move 계열의 함수
    XPathNavigator는 DOM 모델에서 노드 단위로 이동하는 함수를 제공한다.
  - Compile()
    XPath 쿼리를 만드는데 필요한 함수를 제공한다.
  - Select()
    XPath의 쿼리에 대한 검색 결과를 반환하는 함수를 제공한다.
- **XPathExpression**
  - XPath 쿼리를 구현하는데 필요한 함수를 제공해주는 클래스
- **XPathDocument**
  - IXPathNavigable를 구현
  - DOM 형식 문서에 검색기능을 제공하기 위한 Document

XPathDocument

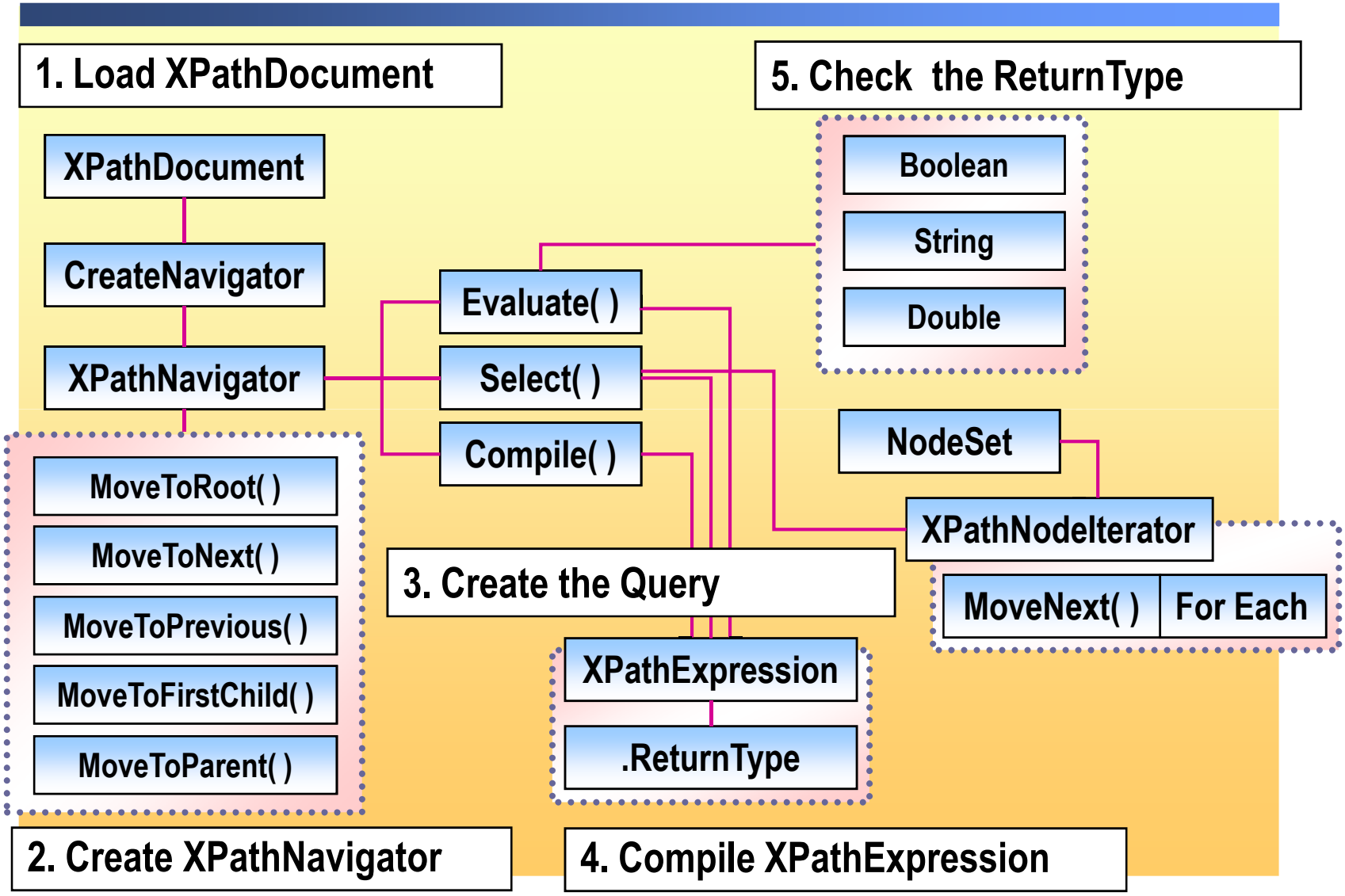
- **XPathDocument 생성**
  - Stream, File, TextReader, XmlReader

```
XPathDocument doc = new XPathDocument(…);
```

- **XPathDocument 검색**

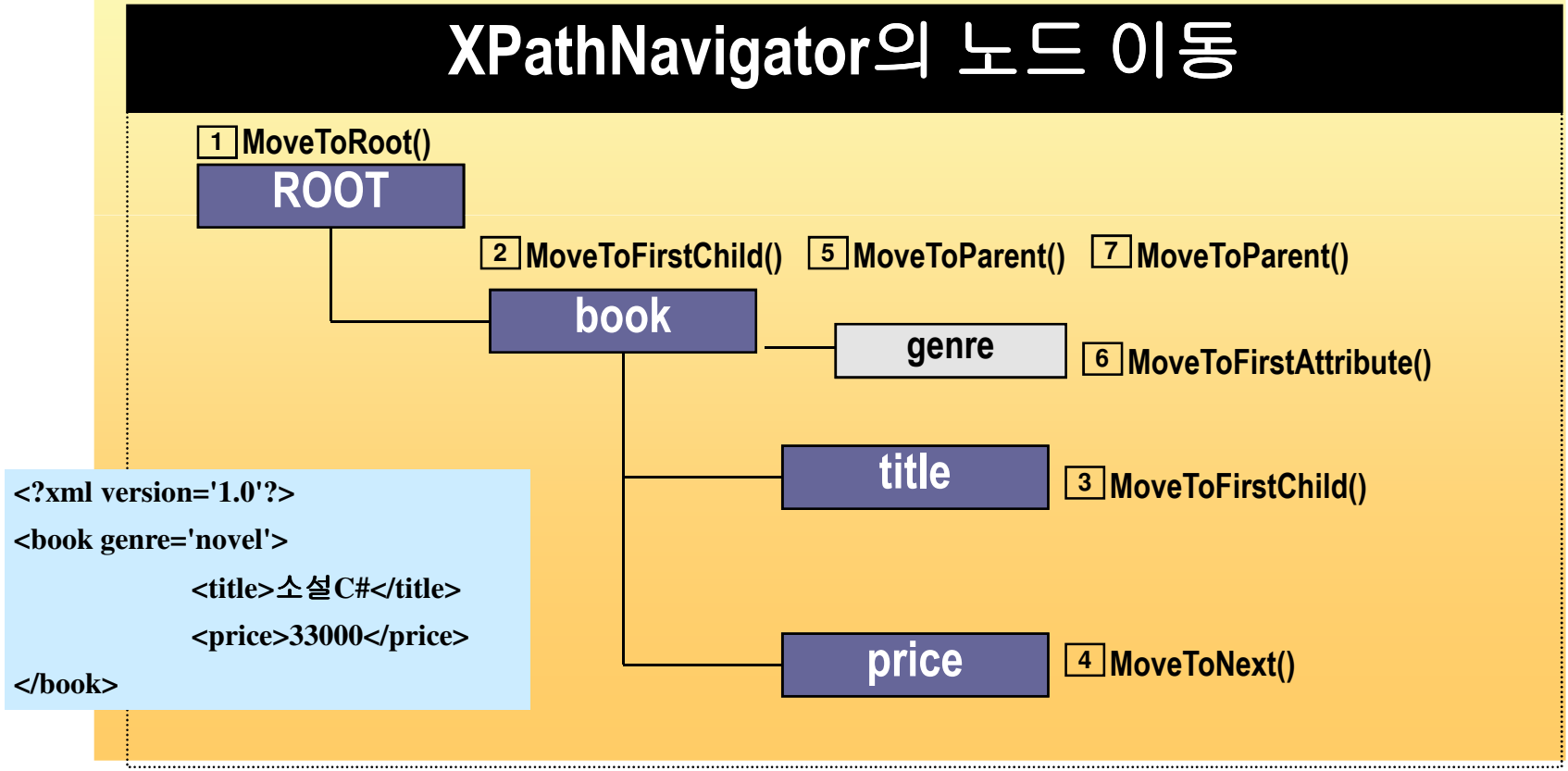
Method	Action
MoveToRoot	Moves the navigator to the root of the document
MoveToNext	Moves the navigator to the next sibling of the current node
MoveToPrevious	Moves the navigator to the previous sibling of the current node
MoveToFirstChild	Moves the navigator to the first child of a node
MoveToParent	Moves the navigator to the parent of the current node

쿼리 과정


1. Load XPathDocument
XPathDocument
CreateNavigator
XPathNavigator
MoveToRoot( )
MoveToNext( )
MoveToPrevious( )
MoveToFirstChild( )
MoveToParent( )
2. Create XPathNavigator
Evaluate( )
Select( )
Compile( )
3. Create the Query
XPathExpression
.ReturnType
4. Compile XPathExpression
5. Check the ReturnType
Boolean
String
Double
NodeSet
XPathNodeIterator
MoveNext( )
For Each

◆XPathNavigator의 노드 검색

- **XPathNavigator 생성하는 방법**
  - XPathDocument xpD = new XPathDocument("파일이름");



XPathNavigator를 이용한 재귀적 노드 검색

```
public class XPathNavigatorTest2{
    public static void Main()  {
            XPathDocument xpD = new XPathDocument("sample.xml");
            XPathNavigator xpN = xpD.CreateNavigator();
            xpN.MoveToRoot();
            DisplayTree(xpN);
    }
    public static void DisplayTree (XPathNavigator xpN){
            if (xpN.HasChildren){
                    xpN.MoveToFirstChild();
                    DisplayInfo (xpN);
                    DisplayTree (xpN);
                    xpN.MoveToParent();
            }
            while (xpN.MoveToNext()){
                    DisplayInfo (xpN);
                    DisplayTree (xpN);
            }
    }
}
```

XPathNavigator의 Select()

- **XPathNavigator의 Select() 함수 호출 I**

```
XPathDocument doc = new XPathDocument("sample.xml");
XPathNavigator nav = doc.CreateNavigator();
XPathNodeIterator iter = nav.Select("bookstore/book/title");
while(iter.MoveNext())
        Console.WriteLine(iter.Current.Value);
```

- **XPathNavigator의 Select() 함수 호출 II**

```
XPathDocument doc = new XPathDocument("sample.xml");
XPathNavigator nav = doc.CreateNavigator();
XPathExpression expr = nav.Compile("bookstore/book/title");
XPathNodeIterator iter = nav.Select(expr);
while(iter.MoveNext())
        Console.WriteLine(iter.Current.Value);
```

XPathNavigator의 Matches()

- **XPathNavigator로 검색결과 얻어내기**

```
XPathDocument doc = new XPathDocument("sample.xml");
XPathNavigator nav = doc.CreateNavigator();
XPathNodeIterator ni = nav.Select("bookstore/book");
```

- **비교할 XPath식에 대한 XPathExpression**

```
XPathExpression expr = nav.Compile("book[@publicationdate=1991]");
```

- **전체 노드를 순회하면서 Matches()로 비교하기**

```
while (ni.MoveNext()){
    XPathNavigator temp = ni.Current.Clone();
    if (temp.Matches(expr)){
            temp.MoveToFirstChild();//title로 이동
            Console.WriteLine("title:" + temp.Value + " in 1991");
    }
}//while
```

XPathNavigator의 Evaluate()

- **XPathNavigator 생성**

```
XmlDocument doc = new XmlDocument();
doc.Load("sample.xml");
XPathNavigator nav = doc.CreateNavigator();
```

- **XPathExpression 생성**

가격에 해당하는 문자열을 숫자로 변환

```
XPathExpression expr = nav.Compile("sum(bookstore/book/price/text())");
```

합을 구하는 계산식

- **XPathNavigator의 Evaluate()를 이용해서 계산식 처리**

```
double total = (double)nav.Evaluate(expr);
Console.WriteLine("Total price: {0}", total);
```

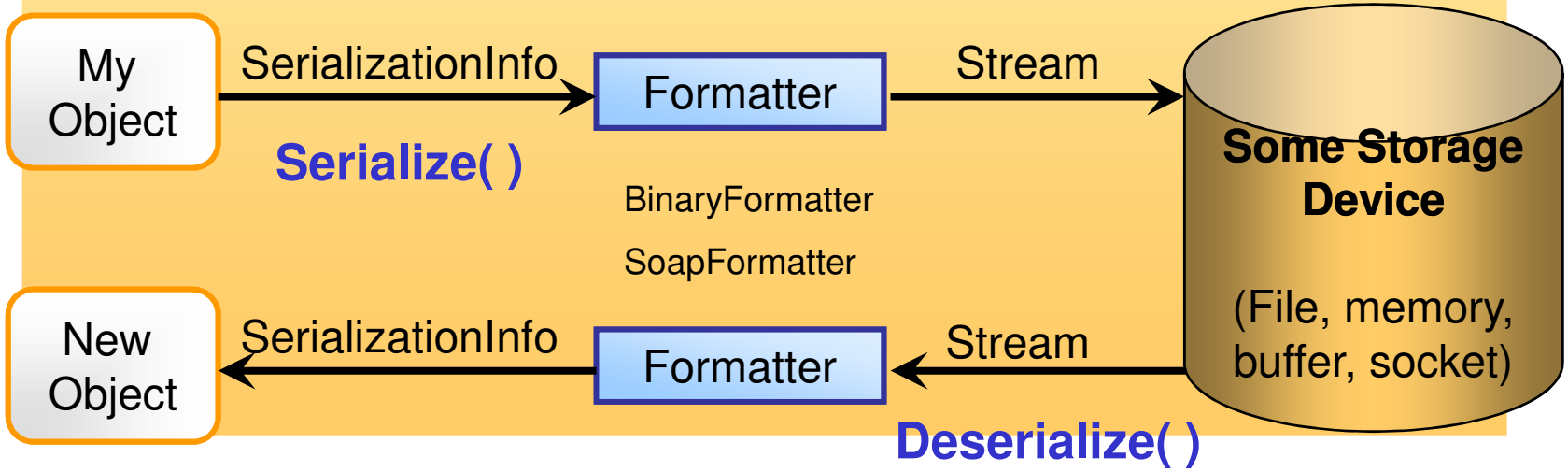
◆XslCompiledTransform를 이용한 문서변환

- **XslCompiledTransform**
  - XSL Transform을 구현하기 위한 클래스
- **파일 구성**
  - XSL 파일 : books.xsl
  - 스키마 파일 : schema.xsd
  - XML 파일: booksXSL2.xml
  - 결과 파일: books.html
- **XslCompiledTransform의 예**
  - XslCompiledTransform xslt = new XslCompiledTransform();
  - xslt.Load("books.xsl");
  - xslt.Transform("booksXSL2.xml", "books.html");

Serialization

- Mark the class with the Serializable attribute

```
[Serializable]
public class MyObject {
  public int    n1 = 0;
  public int    n2 = 0;
  public string str = null;
}
```



Serialization 예제

```
MyObject obj = new MyObject();
obj.n1 = 10;
obj.n2 = 20;
obj.str =“Hello”;

FileStream s = File.Create(“test.bin");
// create the BinaryFormatter
BinaryFormatter b = new BinaryFormatter();
// serialize the graph to the stream
b.Serialize(obj, l);
s.Close();
```

```
// open the filestream
FileStream s = File.OpenRead(“test.bin");
// create the formatter
BinaryFormatter b = new BinaryFormatter();
// deserialize
MyObject p = (MyObject) b.Deserialize(s);
s.Close();
Console.WriteLine(“n1={0} n2={1} str={2}”,p.n1, p.n2,p.str);
```

XML Serialization

- **To serialize and deserialize an object**
  - Create the object
  - Construct an **XMLSerializer**
  - Call the Serialize/Deserialize methods

```
StreamWriter mySWriter = new StreamWriter( @“C:\myFileName.xml" );
try
{
    Serialize the MyObject object
   XmlSerializer serializer = new XmlSerializer(typeof(MyObject));
   serializer.Serialize( mySWriter, MyObject );
}
finally
{
   mySWriter.Close( );
}
```