

Prepared by Mr. Renato L. Adriano II

XML DOM USING JAVASCRIPT

JavaScript, by default, uses DOM to manipulate HTML.

JavaScript can also be used in manipulating XML inside HTML.

The background of the slide is a watercolor-style splash. It features a large, irregular shape in the center. The top half of this shape is a vibrant orange, while the bottom half transitions into a deep blue. The edges of the splash are soft and feathered, with numerous small droplets and splatters of the orange and blue colors scattered across the white background.

SAMPLE XML DOM

```
<body>
  <xml id="xmldata" style="display:none;">
    <students>
      <student id="20180602">
        <name>John Dela Cruz</name>
        <birthday>January 1, 1998</birthday>
        <course>BSIT</course>
      </student>
      <student id="20190818">
        <name>Jane Santos</name>
        <birthday>June 12, 1999</birthday>
        <course>BSIT</course>
      </student>
      <student id="20170408">
        <name>Ryan Reyes</name>
        <birthday>May 8, 1997</birthday>
        <course>BSIT</course>
      </student>
      <student id="20160701">
        <name>Carlo Ople</name>
        <birthday>July 1, 1996</birthday>
        <course>BSIT</course>
      </student>
    </students>
  </xml>
</body>
```

STEP 1: CREATE XML DATA IN HTML

- To add XML data in HTML, use the <xml> tag.
- Provide an id for the <xml> tag.
- Set the style to display:none to hide the XML to the users.

```
<head>
  <script>
    function getData()
    {
      xml = document.getElementById("xmldata");

      studentNames = xml.getElementsByTagName("name");

      document.write("LIST OF STUDENTS<br><br>");

      for(studentName of studentNames)
      {
        document.write(studentName.nodeName + ": ");
        document.write(studentName.childNodes[0].nodeValue + "<br>");
        document.write("<br>");
      }
    }
  </script>
</head>
```

STEP 2A: CREATE JAVASCRIPT FUNCTION

- Create a JavaScript function that is set to manipulate the XML data in the HTML.

NOTE: JavaScript could be internal or external.

```
<head>
  <script>
    function getData()
    {
      xml = document.getElementById("xmldata");

      studentNames = xml.getElementsByTagName("name");

      document.write("LIST OF STUDENTS<br><br>");

      for(i=0;i<4;i++)
      {
        document.write(studentNames[i].nodeName + ": ");
        document.write(studentNames[i].childNodes[0].nodeValue + "<br>");
        document.write("<br>");
      }
    }
  </script>
</head>
```

STEP 2B: CREATE JAVASCRIPT FUNCTION

- Create a JavaScript function that is set to manipulate the XML data in the HTML.

NOTE: JavaScript could be internal or external.

```
    </script>
</head>
<body onload="getData();" >
    <xml id="xmldata" style="display:none;">
        <students>
            <student id="20180602">
```

STEP 3: TRIGGER EVENT TO ACCESS XML DATA

- Add an event to any tags on your HTML, except for the <xml> tag.
- onload event is added to the <body> tag to manipulate the XML data.
- Set the value of the onload event to getData(), the function created in JavaScript.

OUTPUT

- After setting the onload event, run the code using any browser.

LIST OF STUDENTS

NAME: John Dela Cruz

NAME: Jane Santos

NAME: Ryan Reyes

NAME: Carlo Ople

An abstract watercolor splash background. The central part is a large, irregular orange shape that fades into a blue shape at the bottom. The edges are splattered with paint, creating a textured, artistic look.

DOM PROPERTIES

DOM PROPERTIES

Here are some of the commonly used DOM properties:

- `childNodes`
- `parentNode`
- `nextSibling`
- `nextElementSibling`
- `previousSibling`
- `previousElementSibling`
- `firstChild`
- `firstElementChild`
- `lastChild`
- `lastElementChild`

XML FILE



```
<body onload="getData();">
  <xml id="xmldata" style="display:none;">
    <students>
      <student id="20180602">
        <name>John Dela Cruz</name>
        <birthday>January 1, 1998</birthday>
        <course>BSIT</course>
      </student>
      <student id="20190818">
        <name>Jane Santos</name>
        <birthday>June 12, 1999</birthday>
        <course>BSIT</course>
      </student>
      <student id="20170408">
        <name>Ryan Reyes</name>
        <birthday>May 8, 1997</birthday>
        <course>BSIT</course>
      </student>
      <student id="20160701">
        <name>Carlo Ople</name>
        <birthday>July 1, 1996</birthday>
        <course>BSIT</course>
      </student>
    </students>
  </xml>
</body>
```

childNodes

- returns the children of an element.

```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
```

New line here.
(childNodes[0])

```
xml = document.getElementById("xmldata");
students = xml.getElementsByTagName("student");

document.write("Getting the childNodes of student node.<br><br>");
document.write(students[0].childNodes[1].nodeName);
```

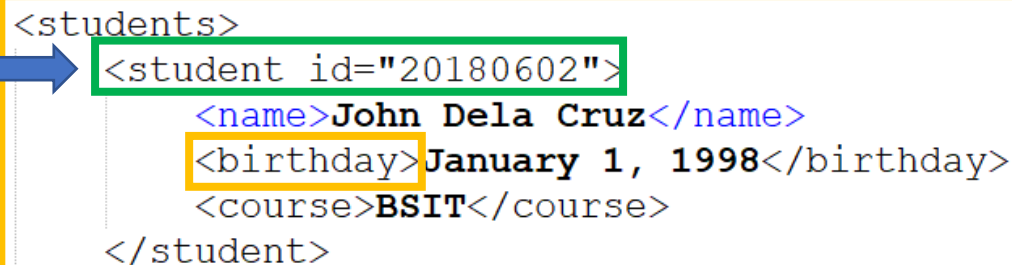
Getting the childNodes of student node.

NAME

parentNode

- returns the parent of an element.

(parentNode)



```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
</students>
```

```
xml = document.getElementById("xmldata");
bdays = xml.getElementsByTagName("birthday");

document.write("Getting the parentNode of birthday node.<br><br>");
document.write(bdays[0].parentNode.nodeName);
```

Getting the parentNode of birthday node.

STUDENT

nextSibling

- returns the next sibling of an element, either if it is an element node or a text node.

```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
```

New line here.
(nextSibling)

```
xml = document.getElementById("xmldata");
names = xml.getElementsByTagName("name");

document.write("Getting the nextSibling of name node.<br><br>");
document.write(names[0].nextSibling.nodeName);
```

Getting the nextSibling of name node.

#text

nextElementSibling

- returns the next element sibling of an element.

```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
```

(nextElementSibling)



```
xml = document.getElementById("xmldata");
names = xml.getElementsByTagName("name");

document.write("Getting the nextElementSibling of name node.<br><br>");
document.write(names[0].nextElementSibling.nodeName);
```


Getting the nextElementSibling of name node.

BIRTHDAY

previousSibling

- returns the previous sibling of an element, either if it is an element node or a text node.

New line here.
(previousSibling)



```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
```

```
xml = document.getElementById("xmldata");
courses = xml.getElementsByTagName("course");

document.write("Getting the previousSibling of course node.<br><br>");
document.write(courses[0].previousSibling.nodeName);
```

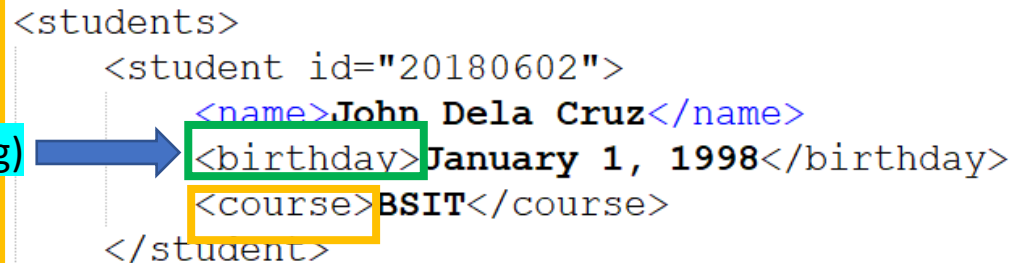
Getting the previousSibling of course node.

#text

previousElementSibling

- returns the previous element sibling of an element.

(previousElementSibling)



```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
</students>
```

```
xml = document.getElementById("xmldata");
courses = xml.getElementsByTagName("course");

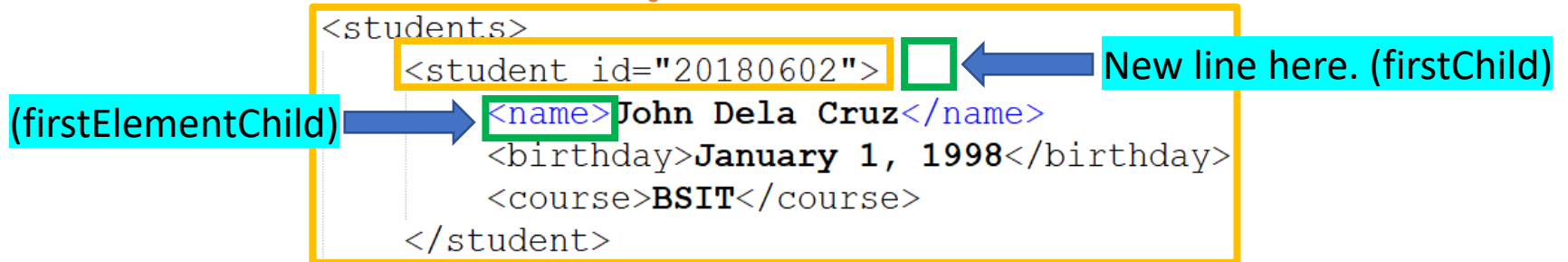
document.write("Getting the previousElementSibling of course node.<br><br>");
document.write(courses[0].previousElementSibling.nodeName);
```

Getting the previousElementSibling of course node.

BIRTHDAY

firstChild/ firstElementChild

- returns the first child node/element child node of the selected element



```
xml = document.getElementById("xmldata");  
students = xml.getElementsByTagName("student");  
  
document.write("Getting the firstChild/ElementChild of student node.<br><br>");  
document.write(students[0].firstChild.nodeName + "<br>");  
document.write(students[0].firstElementChild.nodeName);
```

Getting the firstChild/ElementChild of student node.

#text
NAME

lastChild/ lastElementChild

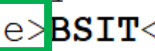
- returns the last child node/element child node of the selected element

```
<students>
  <student id="20180602">
    <name>John Dela Cruz</name>
    <birthday>January 1, 1998</birthday>
    <course>BSIT</course>
  </student>
```

(lastElementChild)



<course>BSIT</course>



</student>



New line here. (lastChild)

```
xml = document.getElementById("xmldata");
students = xml.getElementsByTagName("student");

document.write("Getting the lastChild/ElementChild of student node.<br><br>");
document.write(students[0].lastChild.nodeName + "<br>");
document.write(students[0].lastElementChild.nodeName);
```

Getting the lastChild/ElementChild of student node.

#text

COURSE

The background of the slide is a watercolor-style splash. It features a large, irregular shape in the center, filled with a gradient from bright orange at the top to a deep blue at the bottom. This central shape is surrounded by a lighter, more diffuse area of orange and blue, with small droplets and splatters extending outwards towards the edges of the frame.

DOM NODE PROPERTIES

nodeName

Possible values:

- Returns the tag name for element nodes, in uppercase.
- Returns the name of the attribute for attribute nodes.
- Returns "#text" for text nodes.
- Returns "#comment" for comment nodes.
- Returns "#document" for document nodes.

nodeValue

Possible values:

- Returns null for element nodes and document nodes.
- Returns the value of the attribute for attribute nodes.
- Returns the content for text nodes.
- Returns the content for comment nodes.

nodeType

- nodeType property returns the integer equivalent of the node.

Value	Node Type
1	Element
2	Attribute
3	Text
4	CDATASection
5	EntityReference
6	Entity

nodeType

- nodeType property returns the integer equivalent of the node.

Value	Node Type
7	ProcessingInstruction
8	Comment
9	Document
10	DocumentType
11	DocumentFragment
12	Notation

The background of the slide is a watercolor-style splash. It features a large, irregular shape in the center. The top half of this shape is a vibrant orange, while the bottom half transitions into a deep blue. The edges of the splash are soft and feathered, with smaller droplets and splatters of the same colors scattered across the white background.

NODE TRAVERSING

NODE TRAVERSING

Traversing a Node to Get its Value

- To get the value stored in XML elements, the correct way to traverse or to access is:

Element Node → Text Node → Node Value

NODE TRAVERSING

Selecting the first element with tag name "name".

```
function getData()  
{  
    xml = document.getElementById("xmldata");  
    name1 = xml.getElementsByTagName("name")[0];  
    document.write(name1.childNodes[0].nodeValue);  
}
```

Element Node → Text Node → Node Value

The background of the slide is a watercolor-style splash. It features a large, irregular shape in the center, filled with a gradient from bright orange at the top to a deep blue at the bottom. The edges of this shape are soft and feathered, with smaller, lighter-colored splatters radiating outwards onto the white background.

ACCESSING ATTRIBUTE VALUES

```

<body>
  <xml id="xmldata" style="display:none;">
    <students>
      <student id="20180602">
        <name>John Dela Cruz</name>
        <birthday>January 1, 1998</birthday>
        <course>BSIT</course>
      </student>
      <student id="20190818">
        <name>Jane Santos</name>
        <birthday>June 12, 1999</birthday>
        <course>BSIT</course>
      </student>
      <student id="20170408">
        <name>Ryan Reyes</name>
        <birthday>May 8, 1997</birthday>
        <course>BSIT</course>
      </student>
      <student id="20160701">
        <name>Carlo Ople</name>
        <birthday>July 1, 1996</birthday>
        <course>BSIT</course>
      </student>
    </students>
  </xml>
</body>

```

STEP 1: CREATE XML DATA IN HTML

- To add XML data in HTML, use the <xml> tag.
- Provide an id for the <xml> tag.
- Set the style to display:none to hide the XML to the users.

```
<head>
  <script>
    function getData()
    {
      xml = document.getElementById("xmldata");

      students = xml.getElementsByTagName("student");

      for(student of students)
      {
        document.write("STUDENT NO: ");
        studentNo = student.getAttribute("id");
        document.write(studentNo + "<br>");
      }
    }
  </script>
</head>
```

STEP 2A: CREATE JAVASCRIPT FUNCTION

- Create a JavaScript function that is set to manipulate the XML data in the HTML.

NOTE: JavaScript could be internal or external.

```
<head>
  <script>
    function getData()
    {
      xml = document.getElementById("xmldata");

      students = xml.getElementsByTagName("student");

      for(i=0;i<4;i++)
      {
        document.write("STUDENT NO: ");
        studentNo = students[i].getAttribute("id");
        document.write(studentNo + "<br>");
      }
    }
  </script>
</head>
```

STEP 2B: CREATE JAVASCRIPT FUNCTION

- Create a JavaScript function that is set to manipulate the XML data in the HTML.

NOTE: JavaScript could be internal or external.

```
    </script>
</head>
<body onload="getData();" >
    <xml id="xmldata" style="display:none;">
        <students>
            <student id="20180602">
```

STEP 3: TRIGGER EVENT TO ACCESS XML DATA

- Add an event to any tags on your HTML, except for the <xml> tag.
- onload event is added to the <body> tag to manipulate the XML data.
- Set the value of the onload event to getData(), the function created in JavaScript.

OUTPUT

- After setting the onload event, run the code using any browser.

```
STUDENT NO: 20180602
STUDENT NO: 20190818
STUDENT NO: 20170408
STUDENT NO: 20160701
```



LIST OF STUDENTS

NAME: John Dela Cruz

BIRTHDAY: January 1, 1998

COURSE: BSIT

NAME: Jane Santos

BIRTHDAY: June 12, 1999

COURSE: BSIT

NAME: Ryan Reyes

BIRTHDAY: May 8, 1997

COURSE: BSIT

NAME: Carlo Ople

BIRTHDAY: July 1, 1996

COURSE: BSIT



END OF LESSON