



COMP 4021

Internet Computing

DOM 2

Dr. Kenneth LEUNG

Slides created by Dr. David ROSSITER

Outline

- This presentation considers the following:
 - Creating and adding nodes to the DOM
 - HTML example
 - SVG example
 - Deleting nodes in the DOM
 - HTML example
 - SVG example
 - Old style DOM code: `document.all`

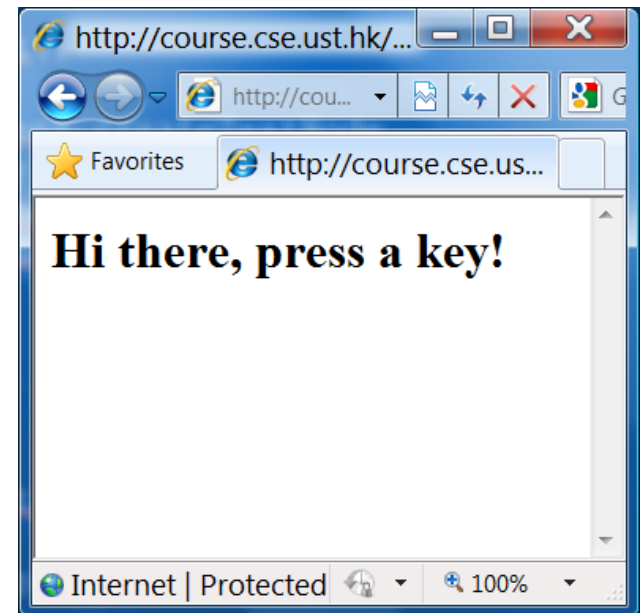
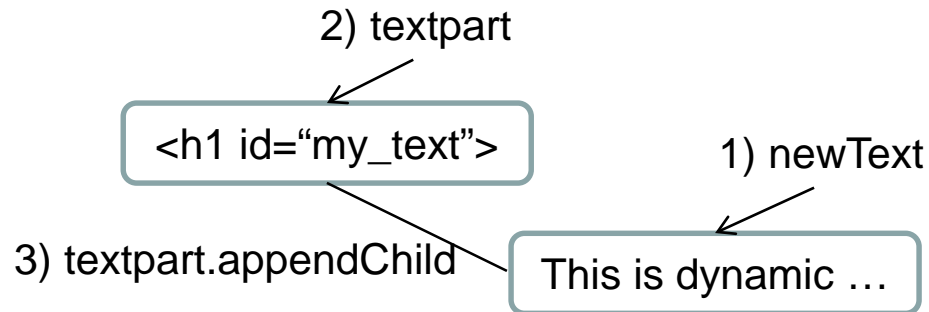
Creating and Adding Nodes to DOM

1. Create a node
2. Add it to the DOM at an appropriate place
 - Right after you created a node (step 1), the node is not actually part of the DOM yet
 - You need to attach it to an existing node in the DOM
 - For visual languages such as HTML and SVG, you won't actually see the node until it is added to the DOM

Dynamic HTML Node Creation – Example

```
<html><head> <script type="text/javascript">
function insert_new_text() {
    var newText = document.createTextNode("This is dynamically added text!");
    var textpart = document.getElementById("my_text");
    textpart.appendChild(newText) ; }    </script>    </head>
```

```
<body onkeypress="insert_new_text()">
<h1 id="my_text" >Hi there, press a key! </h1>
</body>
```



Dynamic Node Creation – SVG Example 1/2

- Dynamic creation of an SVG node

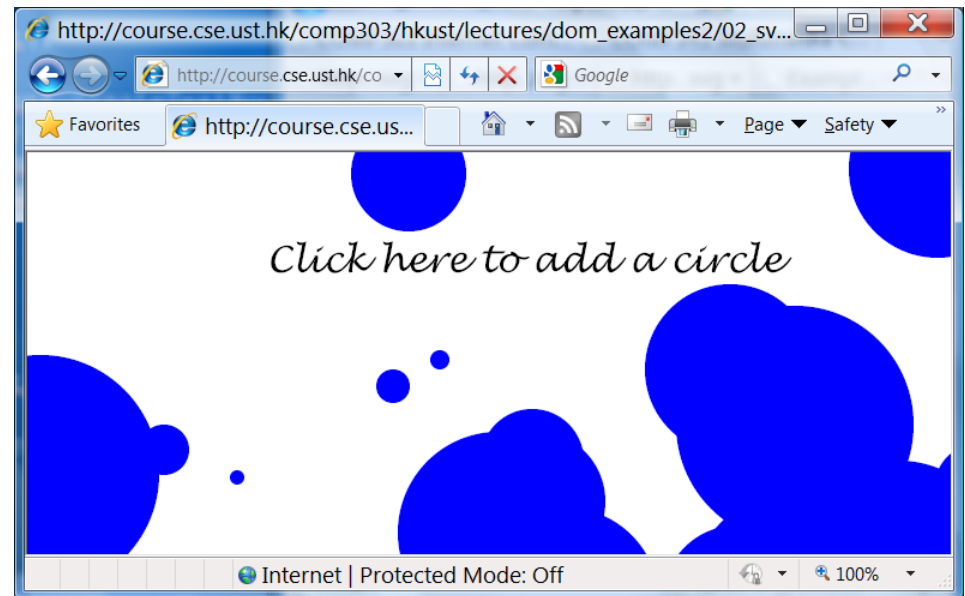
```
<svg width="1000" height="800" onclick="insert_a_circle(evt)" >
```

```
<text x="200" y="100" style="font-size:30px;font-family:Lucida  
Handwriting">
```

Click here to add a circle

```
</text>
```

Example display
after many clicks



Dynamic Node Creation – SVG Example 2/2

```
<script type="text/javascript">
var SVGDocument = null, SVGRoot = null;

function insert_a_circle(event) {
    SVGDocument = event.target.ownerDocument;
    SVGRoot = SVGDocument.documentElement;

    var newnode=SVGDocument.createElementNS(
        "http://www.w3.org/2000/svg","circle");
    var cx=Math.floor(Math.random() * 1000);
    var cy=Math.floor(Math.random() * 800);
    var r=Math.floor(Math.random() * 100);
    newnode.setAttribute('cx', cx);    newnode.setAttribute('cy', cy);
    newnode.setAttribute('r', r);      newnode.setAttribute('fill', "blue");

    SVGRoot.appendChild(newnode); } </script>
```

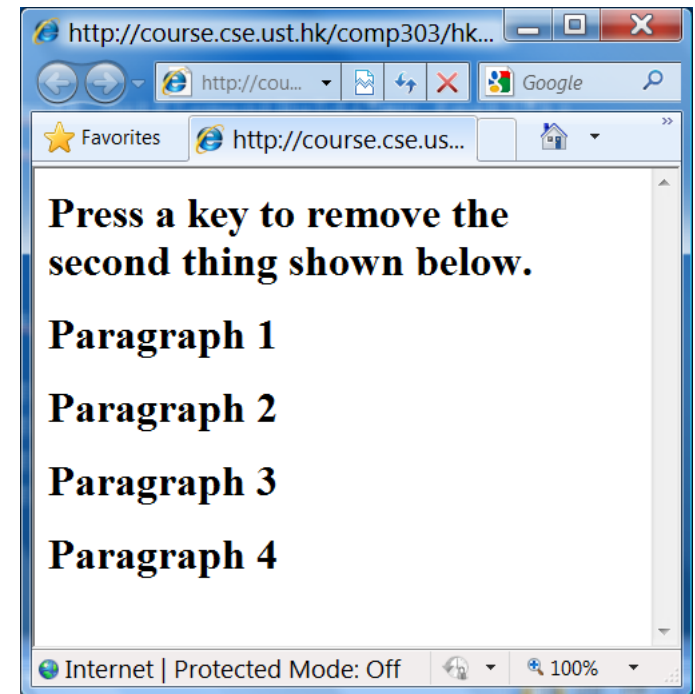
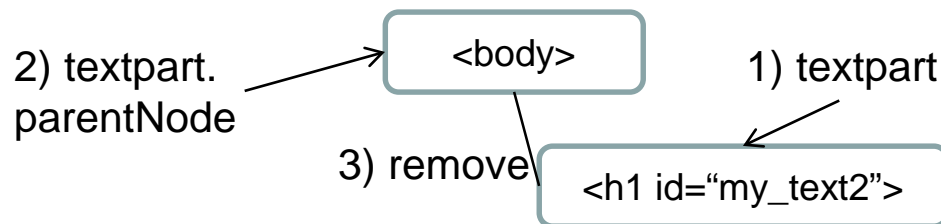
Deleting Nodes

- To delete a node in the DOM, you cannot simply point to a node and say 'delete this'
- Instead, you have to **ask the parent node to delete that child node**
- The parent node may have many children, so you have to specify exactly which child you want the parent to delete

Dynamic Node Deletion – HTML Node

```
function delete_text() {  
    var textpart = document.getElementById("my_text2");  
    textpart.parentNode.removeChild(textpart);  
}
```

```
<body onkeypress="delete_text()">  
<h1 id="my_text1">Paragraph 1</h1>  
<h1 id="my_text2">Paragraph 2</h1>  
<h1 id="my_text3">Paragraph 3</h1>  
<h1 id="my_text4">Paragraph 4</h1>  
</body>
```



- Always deletes the 2nd paragraph; change it to delete the paragraph clicked

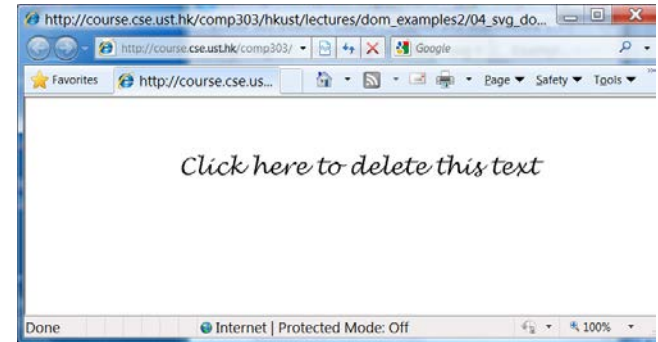
Dynamic Node Deletion – SVG Node

```
<svg width="1000" height="800" onclick="delete_text(evt)">
<script type="text/javascript">
var SVGDocument = null, SVGRoot = null;
var node = null;
```

```
function delete_text(event) {
    SVGDocument = event.target.ownerDocument;

    node = SVGDocument.getElementById("nice_text");
    if (node) node.parentNode.removeChild(node); } </script>
```

```
<text id="nice_text" x="200" y="100"
    style="font-size:30px;font-family:Lucida Handwriting">
    Click here to delete this text</text> </svg>
```



document.all[]

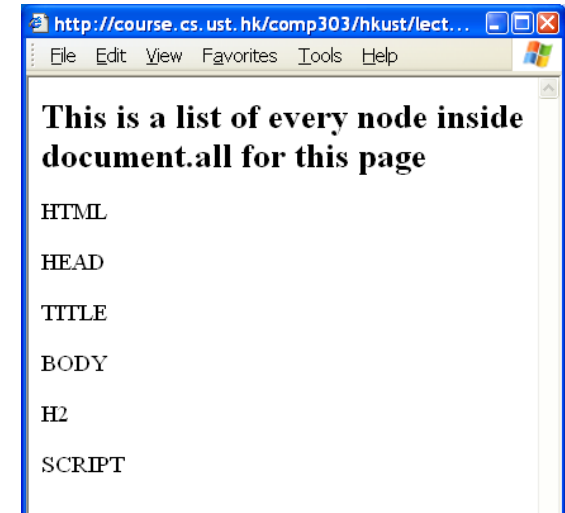
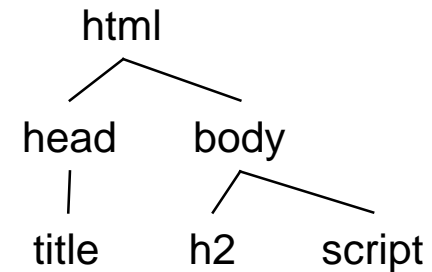
- Another way to access 'anything' in the DOM is by using document.all
 - `document.all["ugly_paragraph"].style.color="black";`
- document.all was created by Microsoft before all the proper DOM existed and is not part of the DOM standard
 - Produce different results in different browsers, and it does not seem to be able to access all nodes in the DOM
 - Please do not use document.all[]
- However, the examples in the next few slides do give further insight into how DOM works dynamically

.all[] Example 1

```
<html> <head><title></title></head>
<body>
<h2>This is a list of every node inside
    document.all for this page</h2>

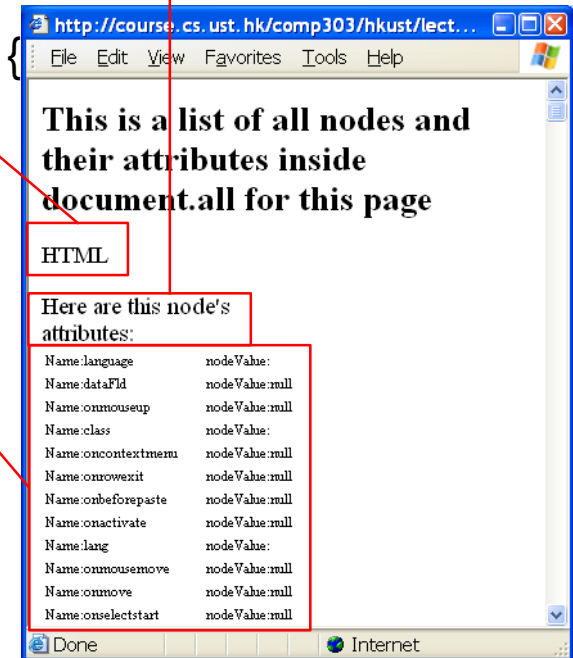
<script language="JavaScript">
var list="";
for (i = 0; i < document.all.length; i++){
    list = list + "<p>" +
        document.all(i).tagName + "</p>";
}
document.write( list );

</script>
</body></html>
```



.all[] Example 2: List tag properties and values

```
for(i = 0; i < document.all.length; i++) {  
    list = list + "<p>" + document.all(i).tagName + "</p>";  
  
    list=list + "<table style='font-size:8pt'><thead>Here are this node's  
    attributes:</thead>";  
  
    for (j=0; j< document.all(i).attributes.length; j++) {  
        list = list + "<tr> <td>Name:" +  
        document.all(i).attributes[j].nodeName +  
        "</td> <td>nodeValue:" +  
        document.all(i).attributes[j].nodeValue +  
        "</td> </tr>";    }  
    list=list + "</table>";    }  
document.write( list );
```



.all[] Example 3: Infinite DOM

```
<html> <head></head>
<body>
<h2>This is an attempt to list every node
inside document.all which cannot succeed</h2>
```

```
<script language="JavaScript">
for (i = 0; i < document.all.length; i++) {
  document.write("<p>" +
    document.all(i).tagName + "</p>");
}
</script> </body> </html>
```

