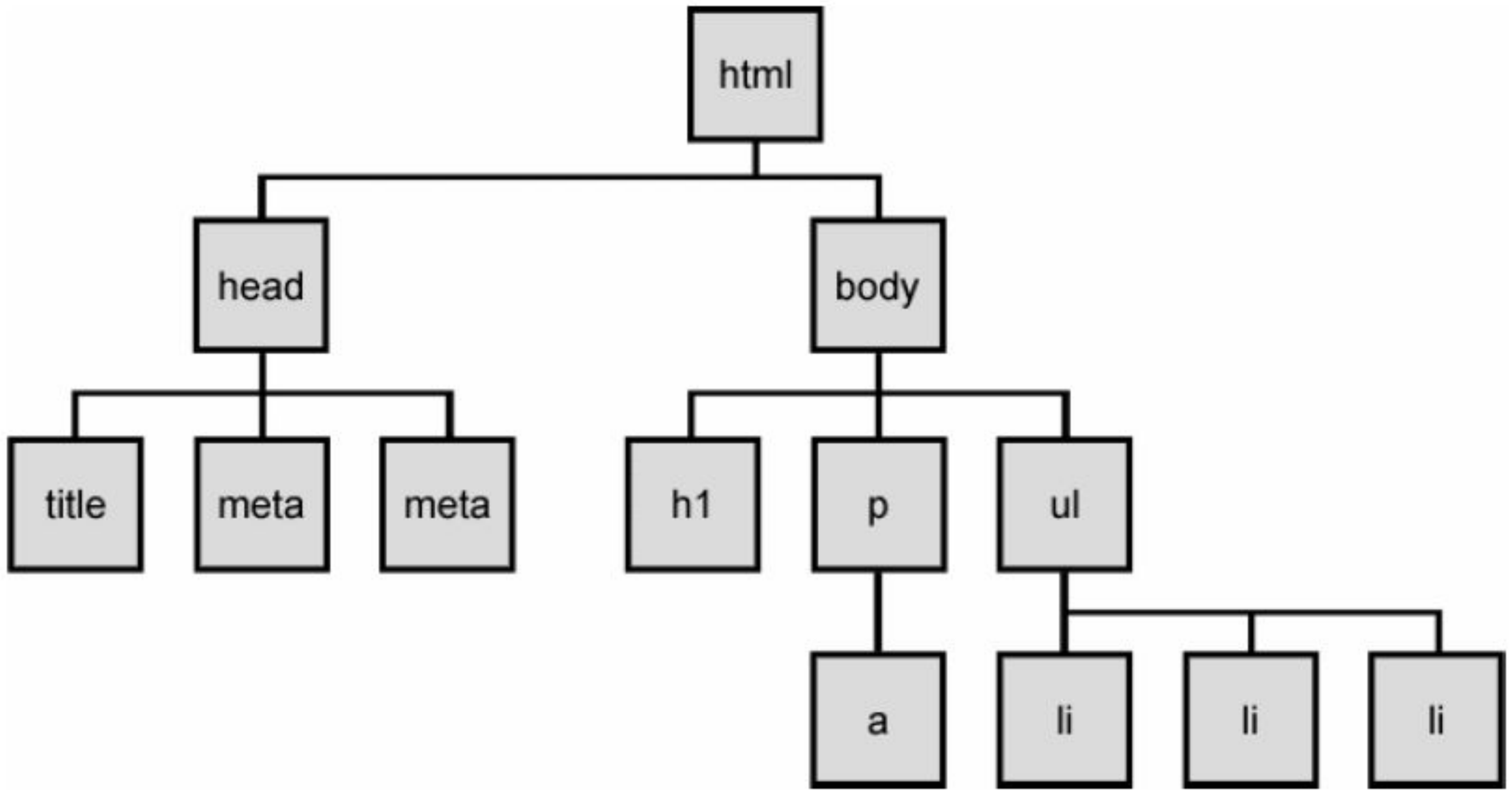
The DOM tree

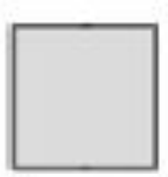
The DOM tree



Types of DOM nodes

```
<p>
This is a paragraph of text with a
<a href="/path/page.html">link in it</a>.
</p>                                          HTML
```

- element nodes (HTML tag)
  - can have children and/or attributes

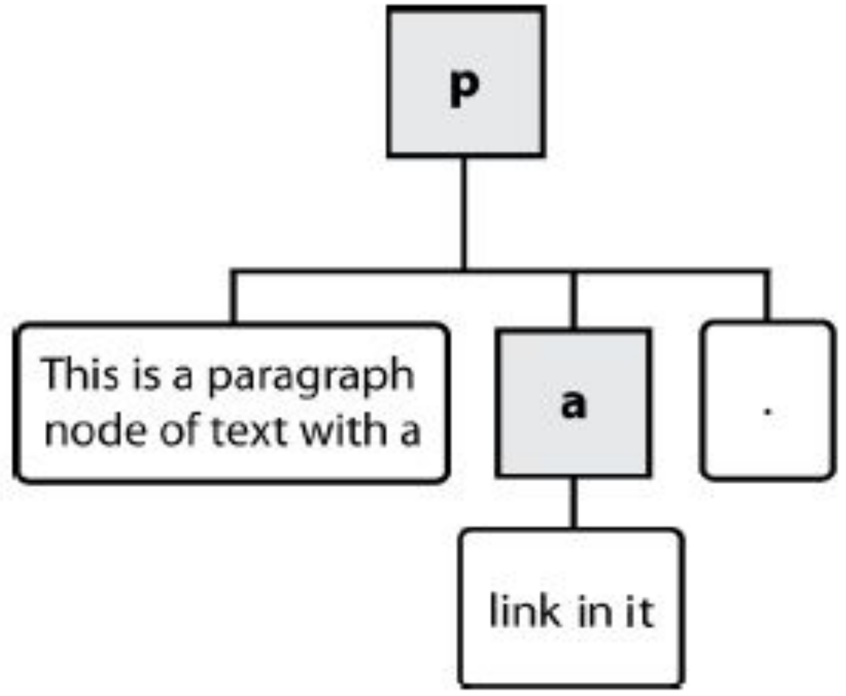


- text nodes (text in a block element)

- attribute nodes (attribute/value pair)
  - text/attributes are children in an element node
  - cannot have children or attributes
  - not usually shown when drawing the DOM tree

Types of DOM nodes

```
<p>
This is a paragraph of text with a
<a href="/path/page.html">link in it</a>.
</p>                                        HTML
```



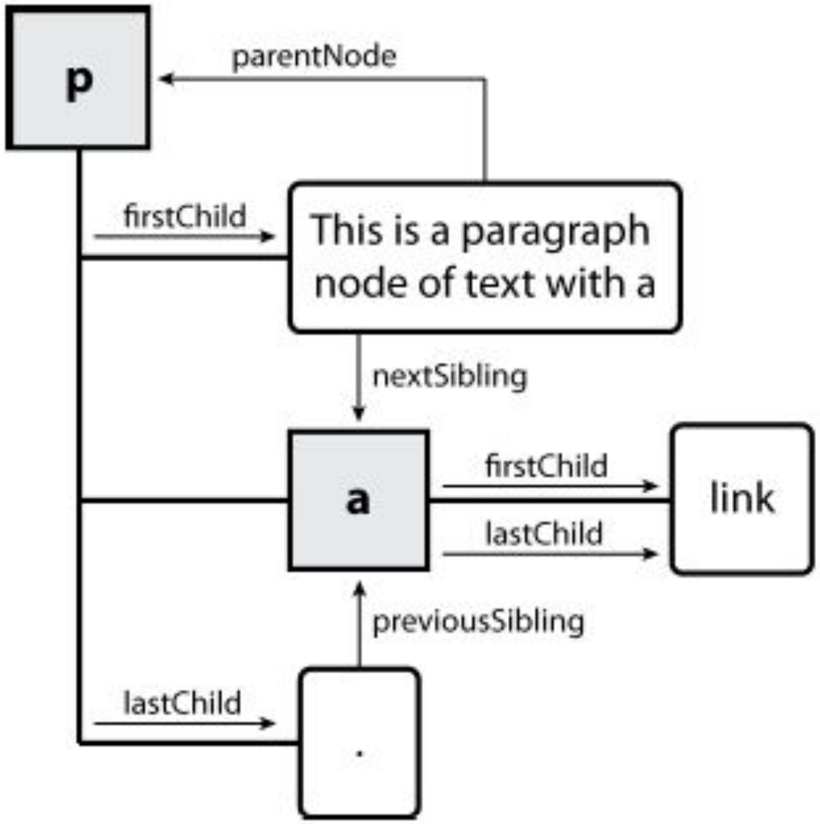
Traversing the DOM tree

name(s)	**description**
firstChild, lastChild	start/end of this node's list of children
childNodes	array of all this node's children
nextSibling, previousSibling	neighboring nodes with the same parent
parentNode	the element that contains this node

DOM tree traversal example

```
<p id="foo">This is a paragraph of text with a
<a href="/path/to/another/page.html">link</a>.</p>
```

HTML



Selecting groups of DOM objects

- methods in document and other DOM objects for accessing descendants:

name	**description**
getElementsByTagName	returns array of descendents with the given tag, such as "div"
getElementsByName	returns array of descendants with the given name attribute (mostly useful for accessing form controls)

Getting all elements of a certain type

```
var allParas = document.getElementsByTagName("p");
for (var i = 0; i < allParas.length; i++) {
    allParas[i].style.backgroundColor = "yellow";
}
```
JS

```
<body>
    <p>This is the first paragraph</p>
    <p>This is the second paragraph</p>
    <p>You get the idea...</p>
</body>
```
HTML

Combining with getElementById

```
var addrParas = $("address").getElementsByTagName("p");
for (var i = 0; i < addrParas.length; i++) {
    addrParas[i].style.backgroundColor = "yellow";
}                                                         JS
```

```
<p>This won't be returned!</p>
<div id="address">
    <p>1234 Street</p>
    <p>Atlanta, GA</p>
</div>                                   HTML
```

The $$ function


```
var arrayName = $$("CSS selector");                    JS
```

```
// hide all "announcement" paragraphs in the "news"
//section
var paragraphs = $$("div#news p.announcement");
for (var i = 0; i < paragraphs.length; i++) {
   paragraphs[i].hide();
}                                            JS
```

- $$ returns an array of DOM elements that match the given CSS selector
  - like $ but returns an array instead of a single DOM object
  - a shorthand for document.select
- useful for applying an operation each one of a set of elements

Common issues with $$


```
// get all buttons with a class of "control"
var gameButtons = ~~$$("control");~~
var gameButtons = $$(".control");                               JS
```

```
// set all buttons with a class of "control" to have red
text
~~$$(".control").style.color = "red";~~
var gameButtons = $$(".control");
for (var I = 0; i < gameButtons.length; i++) {
    gameButtons[i].style.color = "red";
}                                           JS
```

Q: Can I still select a group of elements using $$ even if my CSS file doesn't have any style rule for that same group? (A: Yes!)

Creating new nodes

name	description
document.createElement("tag")	creates and returns a new empty DOM node representing an element of that type
document.createTextNode("text")	creates and returns a text node containing given text

```
// create a new <h2> node
var newHeading = document.createElement("h2");
newHeading.innerHTML = "This is a heading";
newHeading.style.color = "green";
                                JS
```

- merely creating a node does not add it to the page
- you must add the new node as a child of an existing element on the page...

Modifying the DOM tree

name	**description**
appendChild(node)	places given node at end of this node's child list
insertBefore(new, old)	places the given new node in this node's child list just before old child
removeChild(node)	removes given node from this node's child list
replaceChild(new, old)	replaces given child with new node

```
var p = document.createElement("p");
p.innerHTML = "A paragraph!";
$("main").appendChild(p);
                                JS
```

Problems with reading/changing styles

```
window.onload = function() {
   $("clickme").onclick = biggerFont;
};
function biggerFont() {
   var size = parseInt($("clickme").style.fontSize);
   size += 4;
   $("clickMe").style.fontSize = size + "pt";
}                                            JS
```

- style property lets you set any CSS style for an element
- problem: you cannot (usually) read existing styles with it