COMP4021
Internet Computing

Handling the DOM

Gibson Lam and David Rossiter

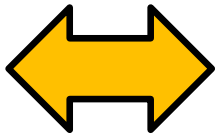
The Document Object Model

- When a web page is loaded in a browser, the page is stored in memory using a tree structure, which is called the DOM (Document Object Model)
- This happens with HTML and also other languages such as XML and SVG (not covered here)
- You can use JavaScript to add, delete or change anything in the DOM structure at any time

An Example DOM Structure

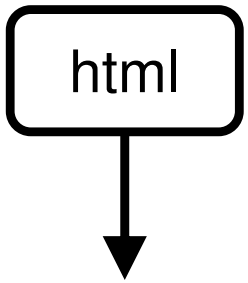
- The DOM is created from the HTML
- The HTML can be created from the DOM

```
<!DOCTYPE html>
<html>
<head>
  <title>Greetings!</title>
</head>
<body>
  <p>How are you?</p>
</body>
</html>
```



The Root Element

- The *root element* of the DOM of a web page is the `<html>` element
- It is at the top of the tree structure
- You can get to the root element using JavaScript, as shown below:




```
let root = document.documentElement;
```

After running the code, the variable points to the top of the DOM, i.e. the <html> element

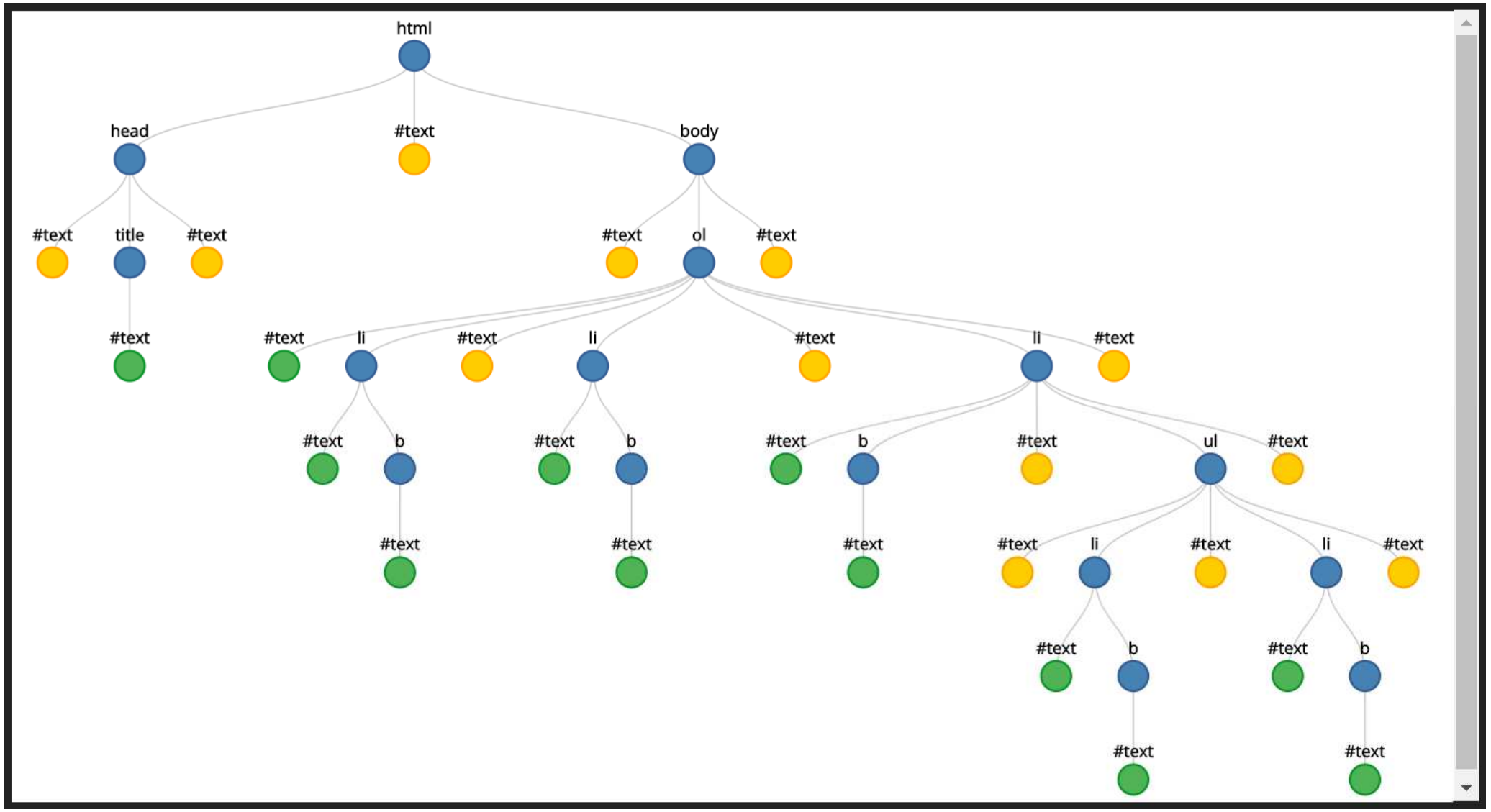
The DOM Visualizer

- Sometimes it is useful to see the DOM structure
- You can use the DOM visualizer here:

http://bioub.github.io/dom-visualizer/

- This visualizer works best for small files
- If you try to see a large file there, you will only messy result

An Example DOM Visualization



DOM Nodes

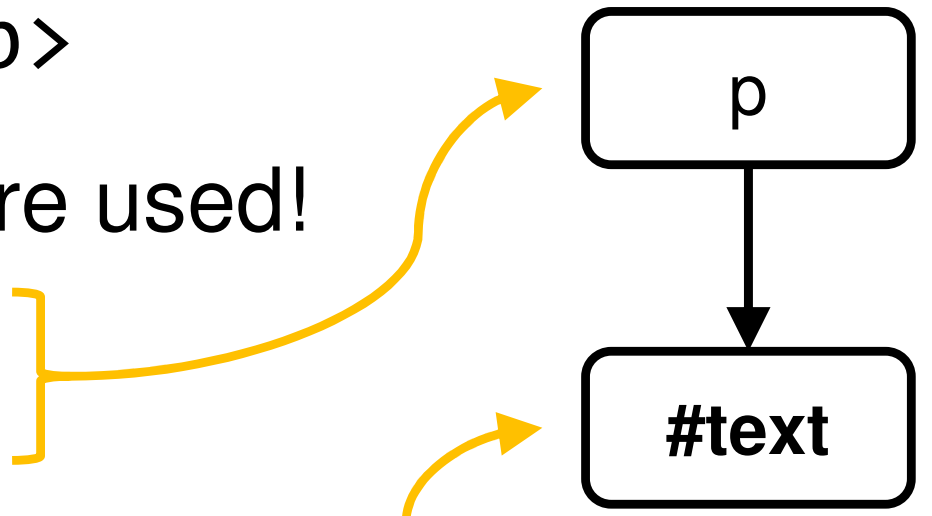
- Every ‘box’ in the DOM tree is called a DOM node
- We will look at two types of DOM nodes:
  - Element nodes e.g. `<p>`
    which store the information of HTML elements
  - Text nodes
    which store text ‘inside’ the element nodes
- There are some other types of node but these are the most common

An Example

- You might think this simple HTML is stored using one node:

```
<p>Hello</p>
```

- However, 2 nodes are used!
- One node is for the HTML element
- Another node is for the text inside it



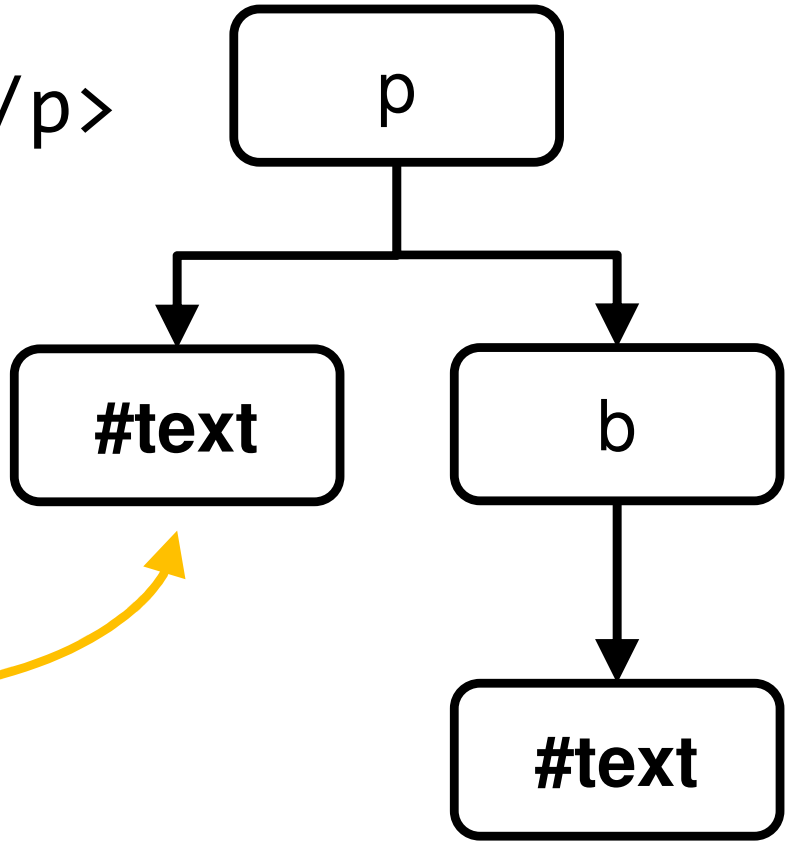
"Hello" is stored inside the text node

Another Example

- Here is an example HTML:

```
<p>Today is <b>Cool</b></p>
```

- It has four nodes
  (two element nodes
  and two text nodes)



"Today is " is stored inside this text node

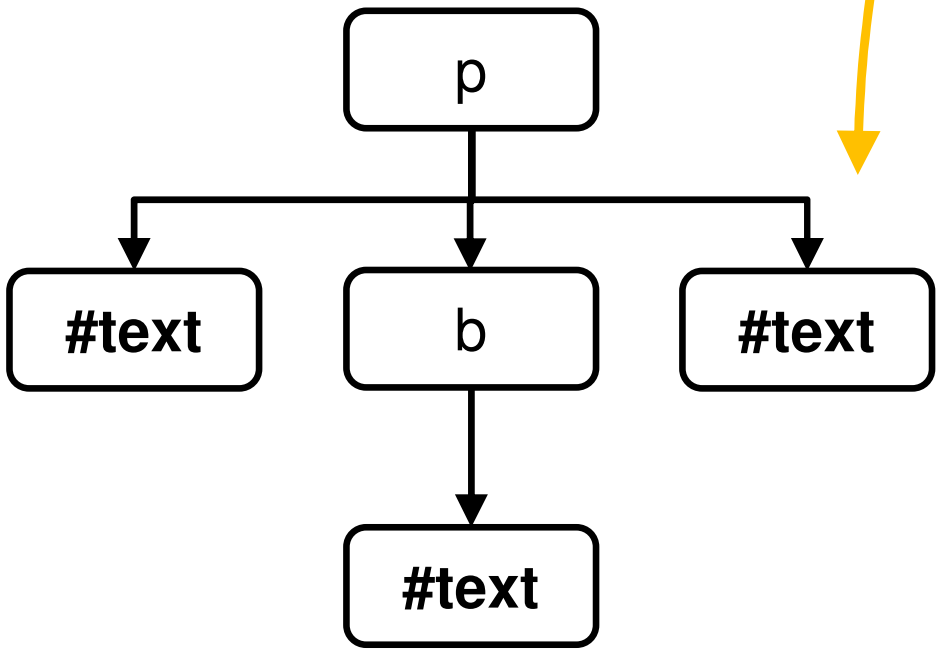
"Cool" is stored inside this text node

Whitespace

- Whitespace means ‘things you can’t see’ in the web page
- Whitespace is also stored in the DOM like everything else
- We usually don’t show whitespace nodes as there are too many of them

```
<p>Today is <b>Cool</b>
</p>
```

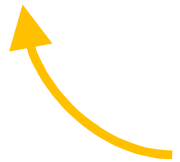
This whitespace is stored in a separate text node



Finding an Element in the DOM

- There are several ways to find an element in the DOM
- One way is to find an element using its `id`
- Here is a reminder of the `id` attribute:

```
<p id="main_text">How are you?</p>
```

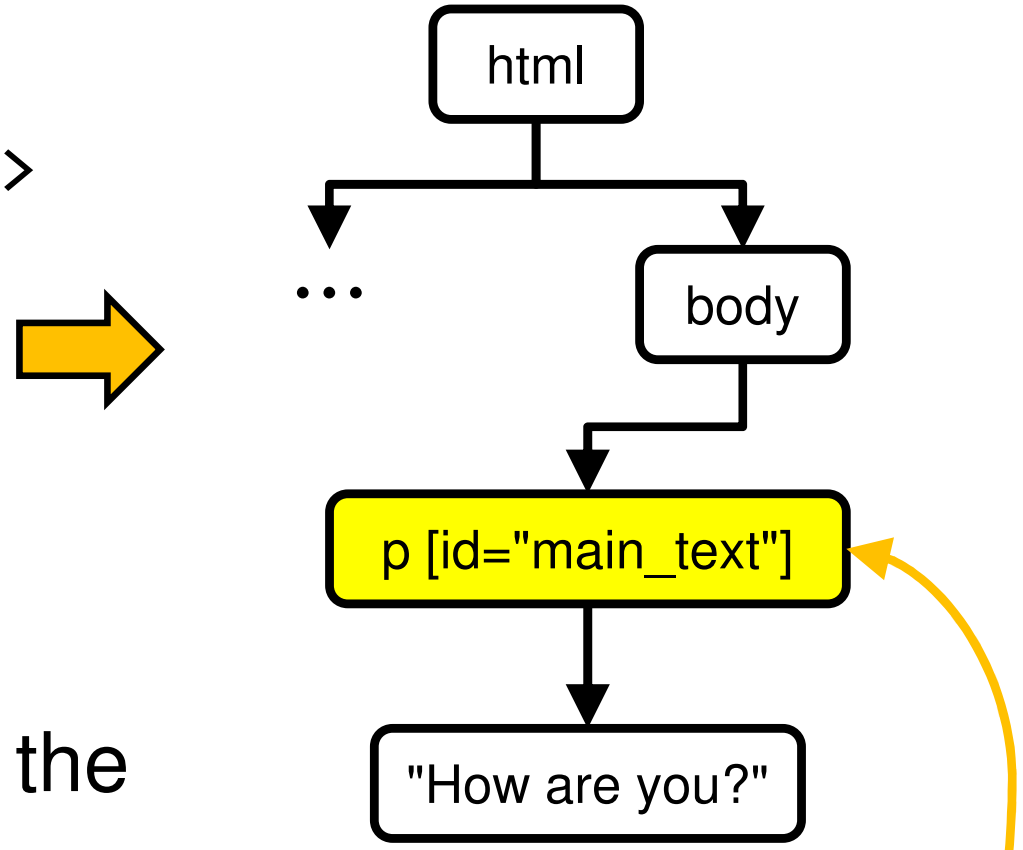


The id of the element is "main_text"

Using getElementById()

- You use `getElementById()` to find an element

```
<body>
     <p id="main_text">
     How are you?
     </p>
     ...
</body>
```



- The following code sets the element to `mynode`

```
mynode = document.getElementById("main_text");
```

Changing the Paragraph Text

- You can change the text content of the paragraph from the previous slide using this code:

```
mynode.innerHTML = "Good morning!";
```

- After running the above code, the HTML would become this:

```
<p id="main_text">Good morning!</p>
```

Changing the Style

- You can also change the style, i.e. CSS properties, of an element node
- For example, if you want to change the background color of an HTML element to red, you can use this code:

```
mynode.style.backgroundColor = "red";
```

This is like:

```
style="background-color: red"
```

The CSS Property Names

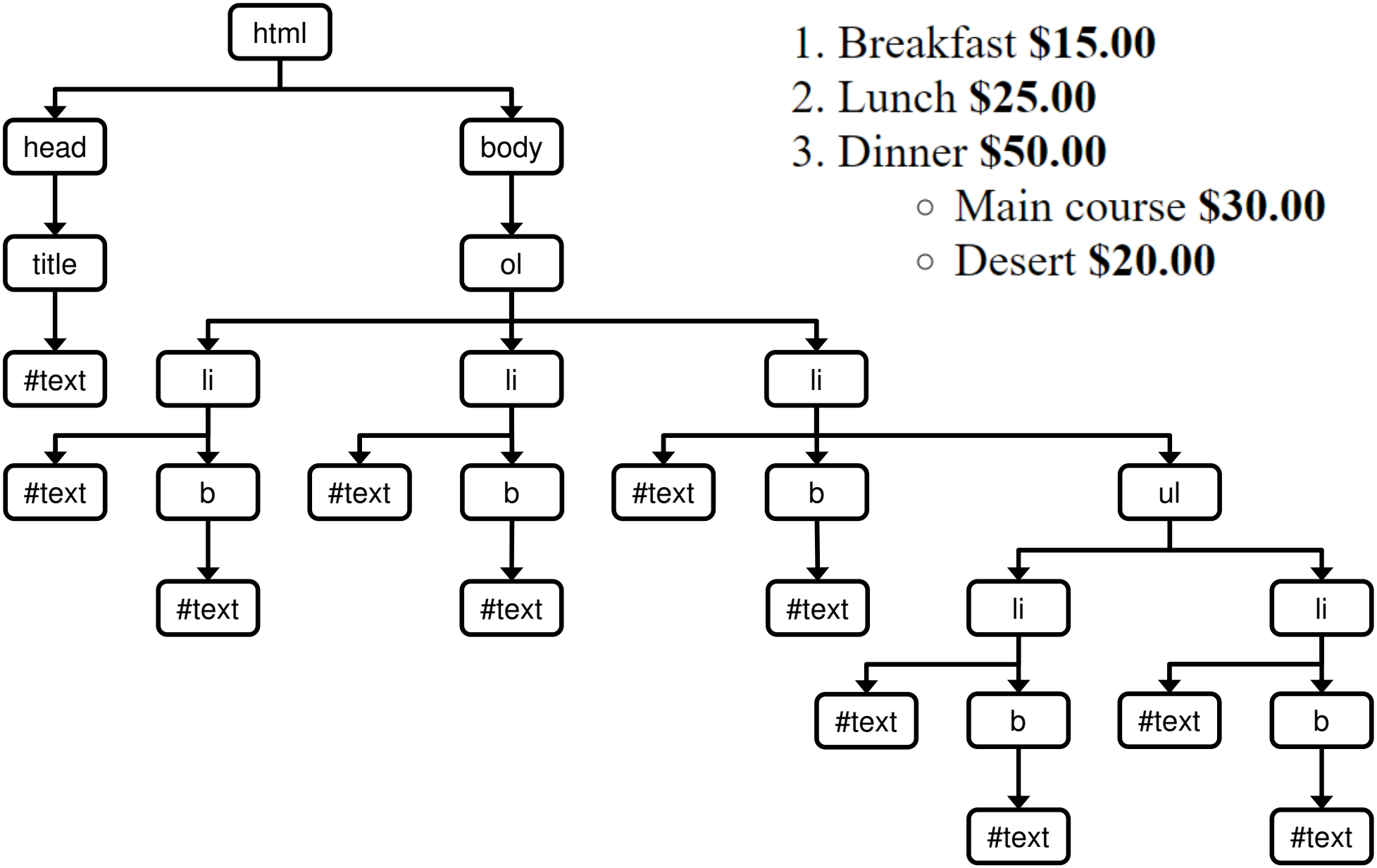
- If the CSS name has a hyphen (`-`) then you need to remove the hyphen and capitalize the following letter
- For example:
  - `background-color` becomes `backgroundColor`
  - `font-family` becomes `fontFamily`

A Bigger Example

```
<!DOCTYPE html>
<html>
<head>
    <title>The Example
            Document</title>
</head>
<body>
    <ol>
        <li>Breakfast <b>$15.00</b></li>
        <li>Lunch <b>$25.00</b></li>
        <li>Dinner <b>$50.00</b>
            <ul>
                <li>Main course <b>$30.00</b></li>
                <li>Desert <b>$20.00</b></li>
            </ul>
        </li>
    </ol>
</body>
</html>
```

1. Breakfast **$15.00**
2. Lunch **$25.00**
3. Dinner **$50.00**
    - Main course **$30.00**
    - Desert **$20.00**

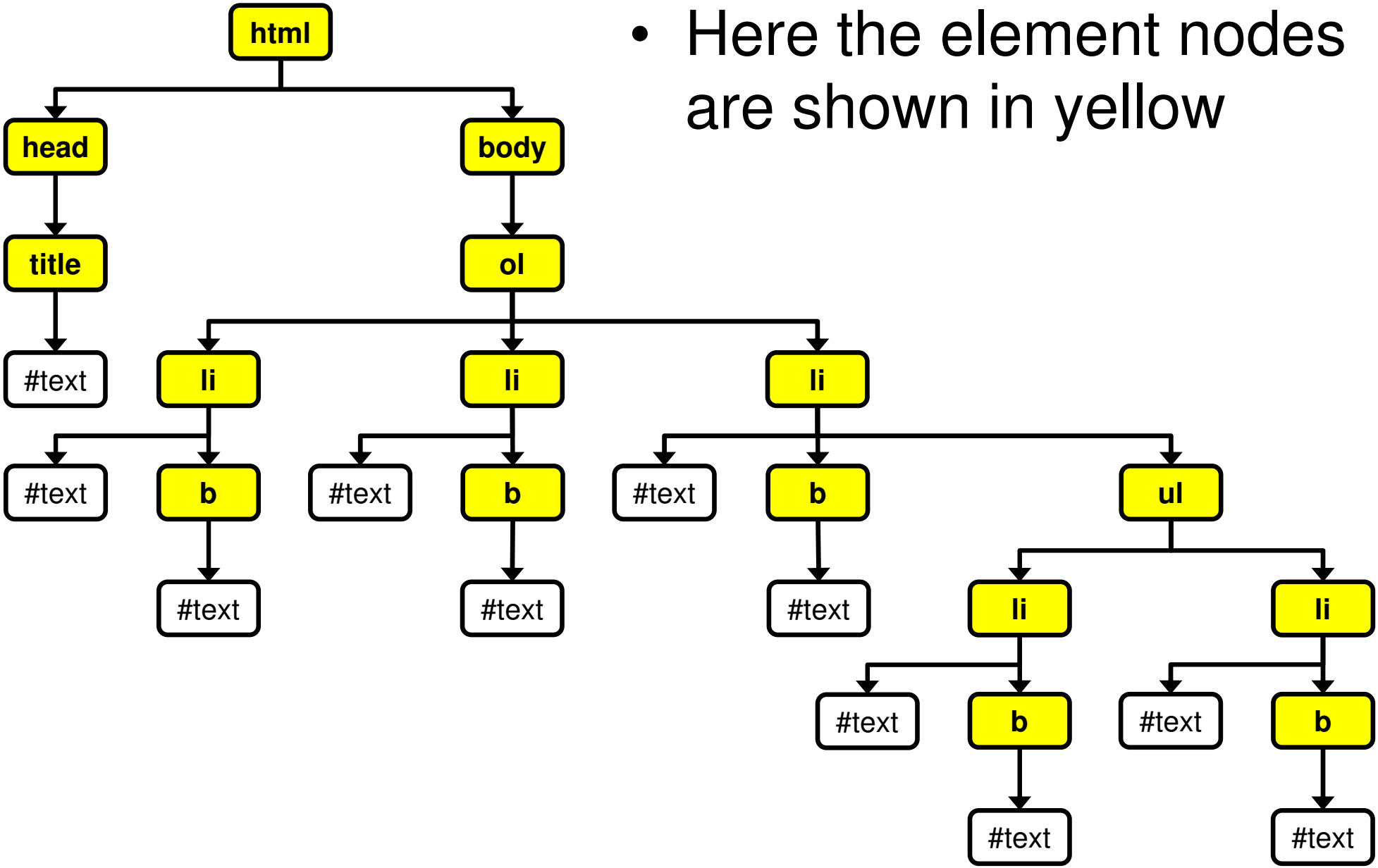
The DOM of the Example



1. Breakfast **$15.00**
2. Lunch **$25.00**
3. Dinner **$50.00**
   - Main course **$30.00**
   - Desert **$20.00**

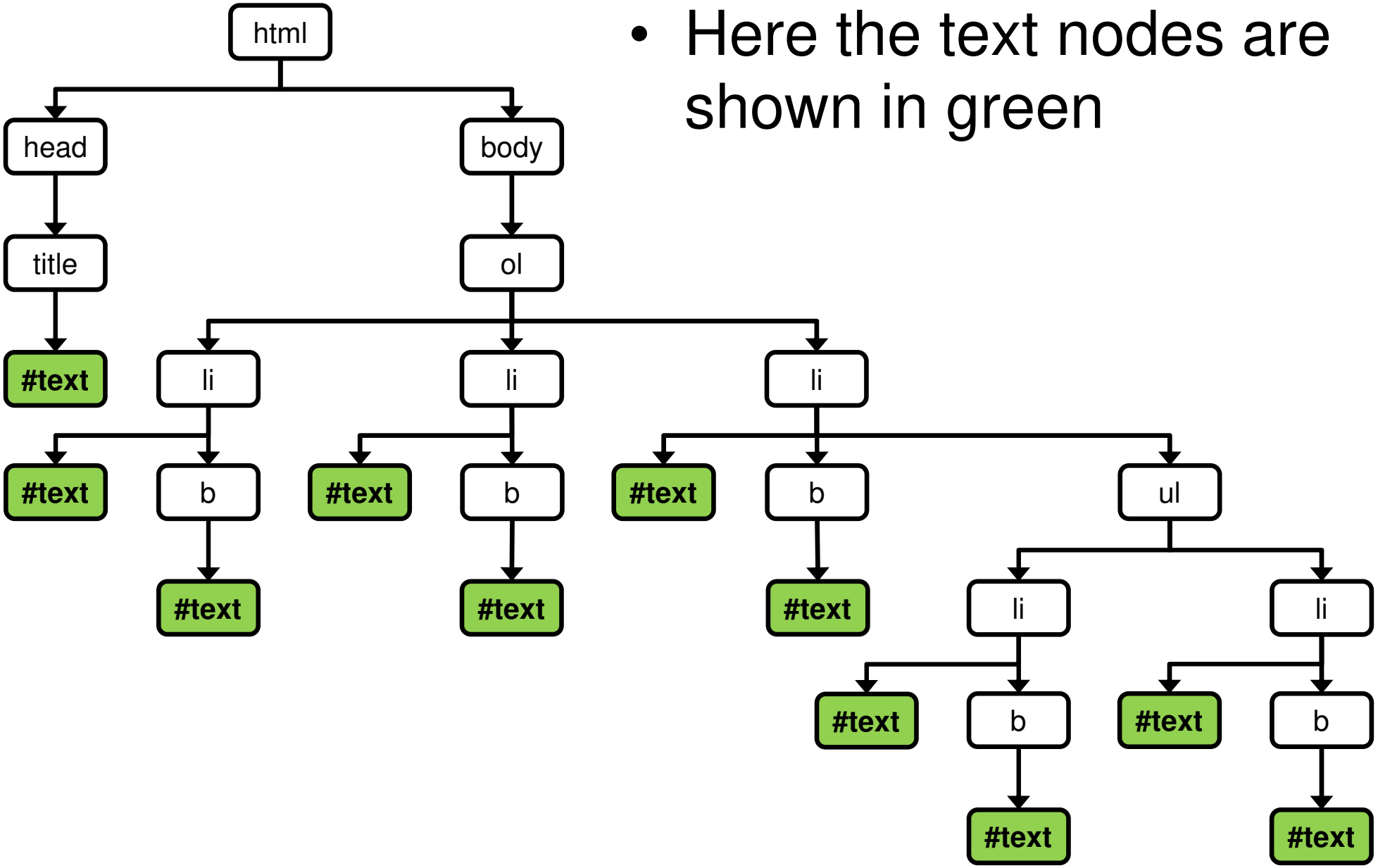
Element Nodes in the Example

- Here the element nodes are shown in yellow



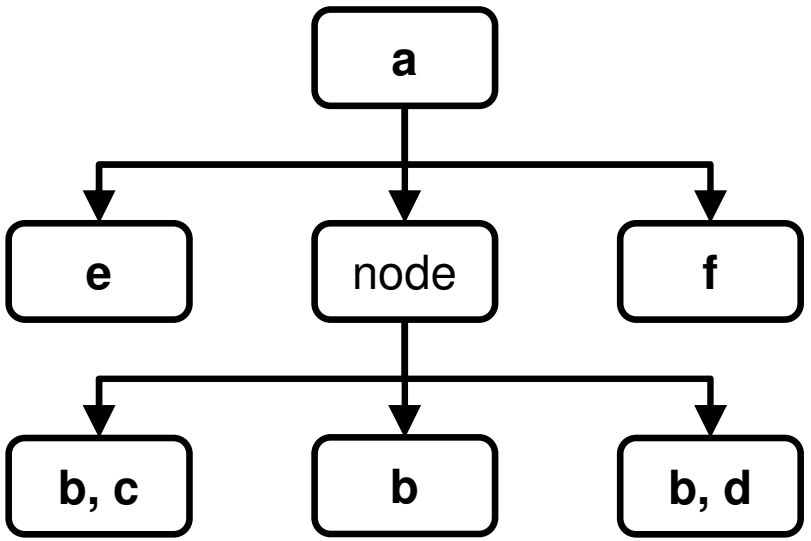
Text Nodes in the Example

- Here the text nodes are shown in green



Traversing the DOM

- You can move around (=traverse) the DOM tree from one node to another
- Here are some useful properties:
  a) `node.parentNode`
  b) `node.childNodes[]`
  c) `node.firstChild`
  d) `node.lastChild`
  e) `node.previousSibling`
  f) `node.nextSibling`



A node can have any number of children

Finding HTML Tags in the DOM

- You can use

  `document.getElementsByTagName()`

  to get all HTML elements which have the same tag, e.g. finding all `<h1>` on the page
- The result is a list containing the matching elements, as shown in the example on the next slide

An Example Using Tag Name

- Here is some JavaScript running with the example HTML, getting all `<li>` nodes:

```
let allLi = document.getElementsByTagName("li");
for (let i = 0; i < allLi.length; i += 2) {
    allLi[i].childNodes[1].style.color = "red";
}
```

- After running the code, three out of the five `<b>` elements are changed to red

1. Breakfast **$15.00**
2. Lunch **$25.00**
3. Dinner **$50.00**
   - Main course **$30.00**
   - Desert **$20.00**