Announcements

- This week
  - No quiz
  - No lecture or labs on Wednesday
  - No school on Thursday or Friday
  - No CLUE Tutoring
- Drop-In labs
  - Tuesday only
  - 8:30-9:20am and 5:00-5:50pm MGH 430

Announcements

- Project 2B due on Wed., December 2

General comments

- The programmer's *habits of mind*
  - Read instructions
  - Work slowly and carefully
  - Pay attention to details

Arrays and conditionals—Setting gender pronouns based on user input…

PROJECT 2B

Overview

- Gender changes based on what the user chose in the dropdown menu
- Arrays set up the series of pronouns for each gender
- Use conditionals to choose which array to use

The gender arrays

```
var MalePronouns = new Array
   ("he","his","him","man","men");
var FemalePronouns = new Array
   ("she","her","her","woman","women");
var PersonPronouns = new Array
   ("one","one's","one","person","persons");
var PeoplePronouns = new Array
   ("they","their","them","people","people");

var gender;
```

Gender in your story

- Each gender has its own pronoun array
  - Edit the array to include the words needed by your story
    - Replace Man with King, Prince, Boy, or Uncle
    - Replace Woman with Queen, Princess, Girl, or Aunt, etc.

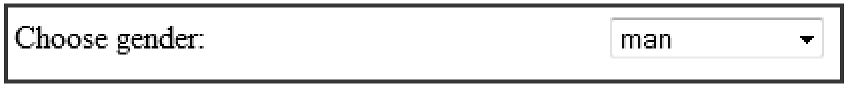
Gender pronoun arrays

- Include only as many pronoun arrays as you have choices in your dropdown menu:

```
var MalePronouns = new Array ("King","he","his");
var FemalePronouns = new Array ("Queen","she","her");
```

Gender dropdown menu

- The user chooses the gender of a main character in the story from the dropdown menu



Choose gender: man

```
<label for="genderlist">Choose gender:</label>
 <select name="genderlist" size="1" id="genderlist">
        <option value="Male">man</option>
         <option value="Female">woman</option>
         <option value="Person">person</option>
         <option value="People">some people</option>
</select>
```

Grab the user's gender choice

- Before you can do anything, you have to grab the user's choice from the genderlist dropdown menu:

var genderlist = **document**.getElementById("genderlist").value;

Grab the user's gender choice

- Before you can do anything, you have to grab the user's choice from the genderlist dropdown menu:

  var genderlist = document.**getElementById(**"genderlist"**)**.value;

Grab the user's gender choice

- Before you can do anything, you have to grab the user's choice from the genderlist dropdown menu:

```
var genderlist = document.getElementById("genderlist").value;
```

Grab the user's gender choice

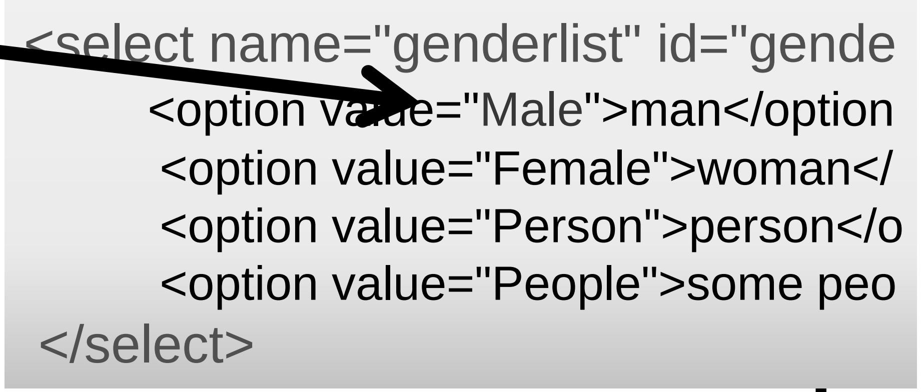
- Before you can do anything, you have to grab the user's choice from the genderlist dropdown menu:

  var genderlist = document.getElementById("genderlist").**value**;

Conditionals

- Conditionals select the pronoun array that matches the gender selected:

```
if (genderlist == "Male")
{
  gender = MalePronouns;
}
else
{
  gender = FemalePronouns;
}
```



```
<select name="genderlist" id="gende
        <option value="Male">man</option
         <option value="Female">woman</
         <option value="Person">person</o
         <option value="People">some peo
 </select>
```

Conditionals

- Conditionals select the pronoun array that matches the gender selected:

```
if (genderlist == "Male")
{
  gender = MalePronouns;
}
else
{
  gender = FemalePronouns;
}
```

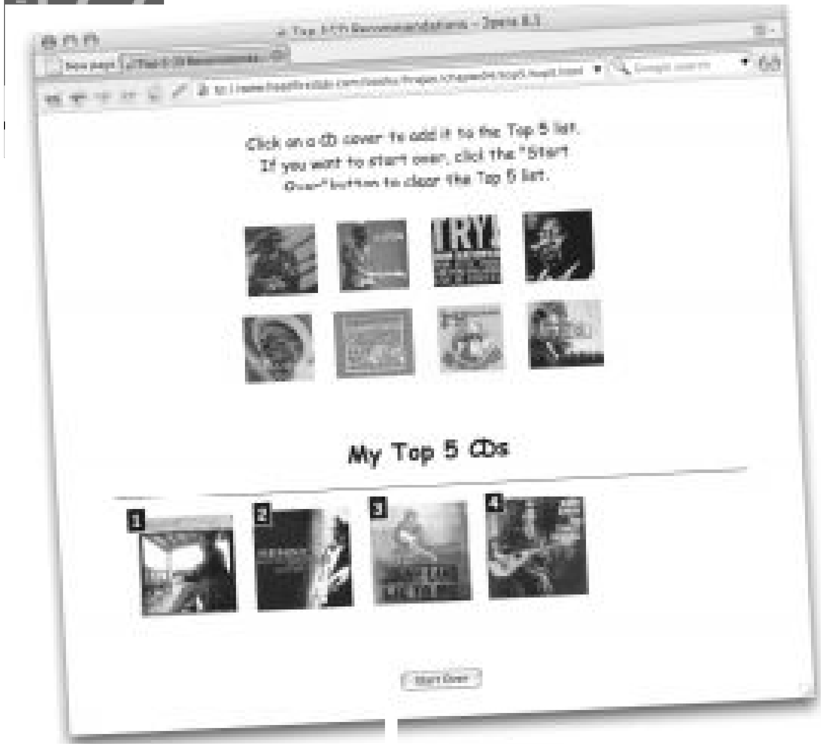
Assigning the array to a variable

- If you assign an array to a variable,
  - *The variable becomes an array*
  - Each element is accessible by array methods

    gender[0] or gender[2]

Understanding the document tree, revisited

DOCUMENT OBJECT MODEL (DOM)



Click on a CD cover to add it to the Top 5 list. If you want to start over, click the "Start Over" button to clear the Top 5 list.

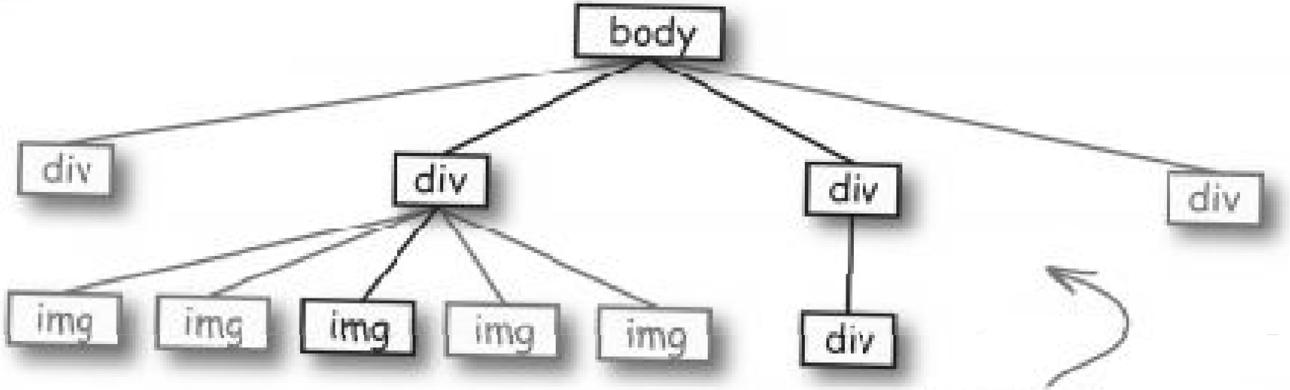
My Top 5 CDs

Use the Document Object Model.

The web browser uses the Document Object Model to represent your web page. When you change this model with your JavaScript code, the web page will automatically change, too.

The DOM lets you look at this web page...

...in this format.



Your JavaScript can work with the DOM much easier than working directly with HTML or CSS.

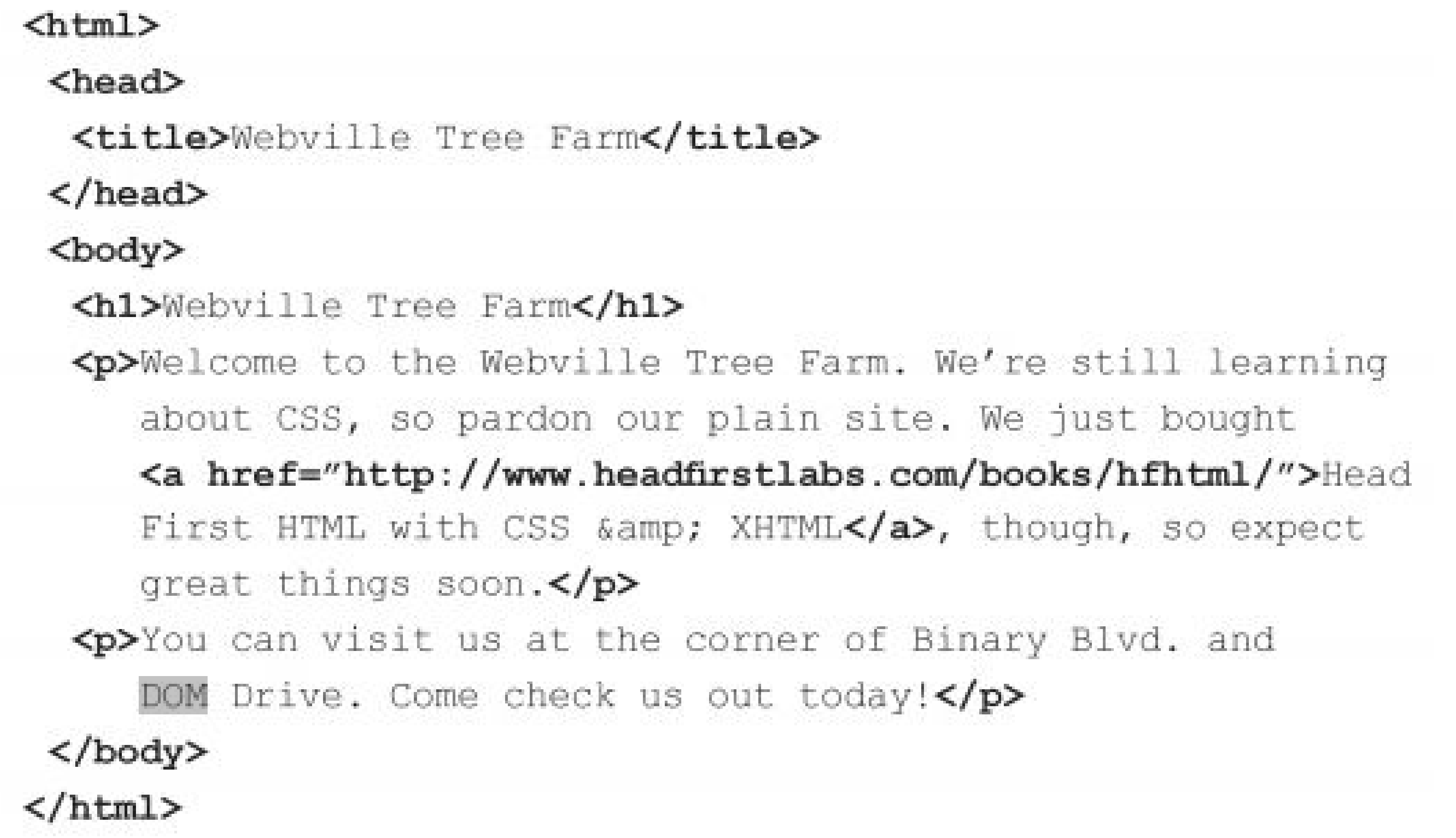
DOM

- The Document object gives you access to the browser's DOM tree
- You've used it for several labs now:

```
document.getElementById("zone1").value
```

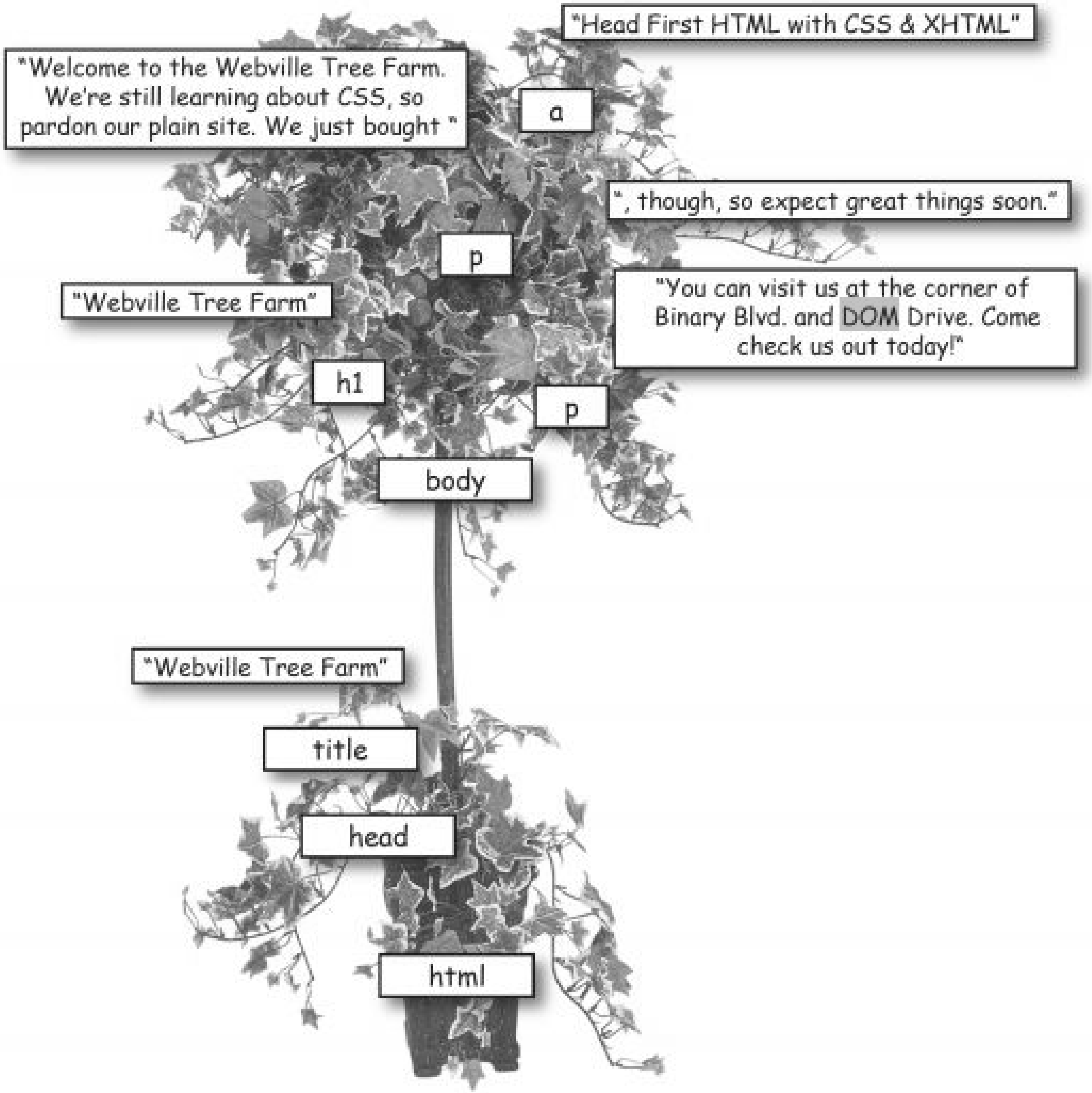
DOM

- Here's the HTML



```
<html>
 <head>
  <title>Webville Tree Farm</title>
 </head>
 <body>
  <h1>Webville Tree Farm</h1>
  <p>Welcome to the Webville Tree Farm. We’re still learning
     about CSS, so pardon our plain site. We just bought
     <a href=”http://www.headfirstlabs.com/books/hfhtml/”>Head
     First HTML with CSS &amp; XHTML</a>, though, so expect
     great things soon.</p>
  <p>You can visit us at the corner of Binary Blvd. and
     DOM Drive. Come check us out today!</p>
 </body>
</html>
```

- As the browser sees it!


"Head First HTML with CSS & XHTML"
"Welcome to the Webville Tree Farm. We're still learning about CSS, so pardon our plain site. We just bought "
a
", though, so expect great things soon."
p
"You can visit us at the corner of Binary Blvd. and DOM Drive. Come check us out today!"
"Webville Tree Farm"
h1
p
body
"Webville Tree Farm"
title
head
html

DOM

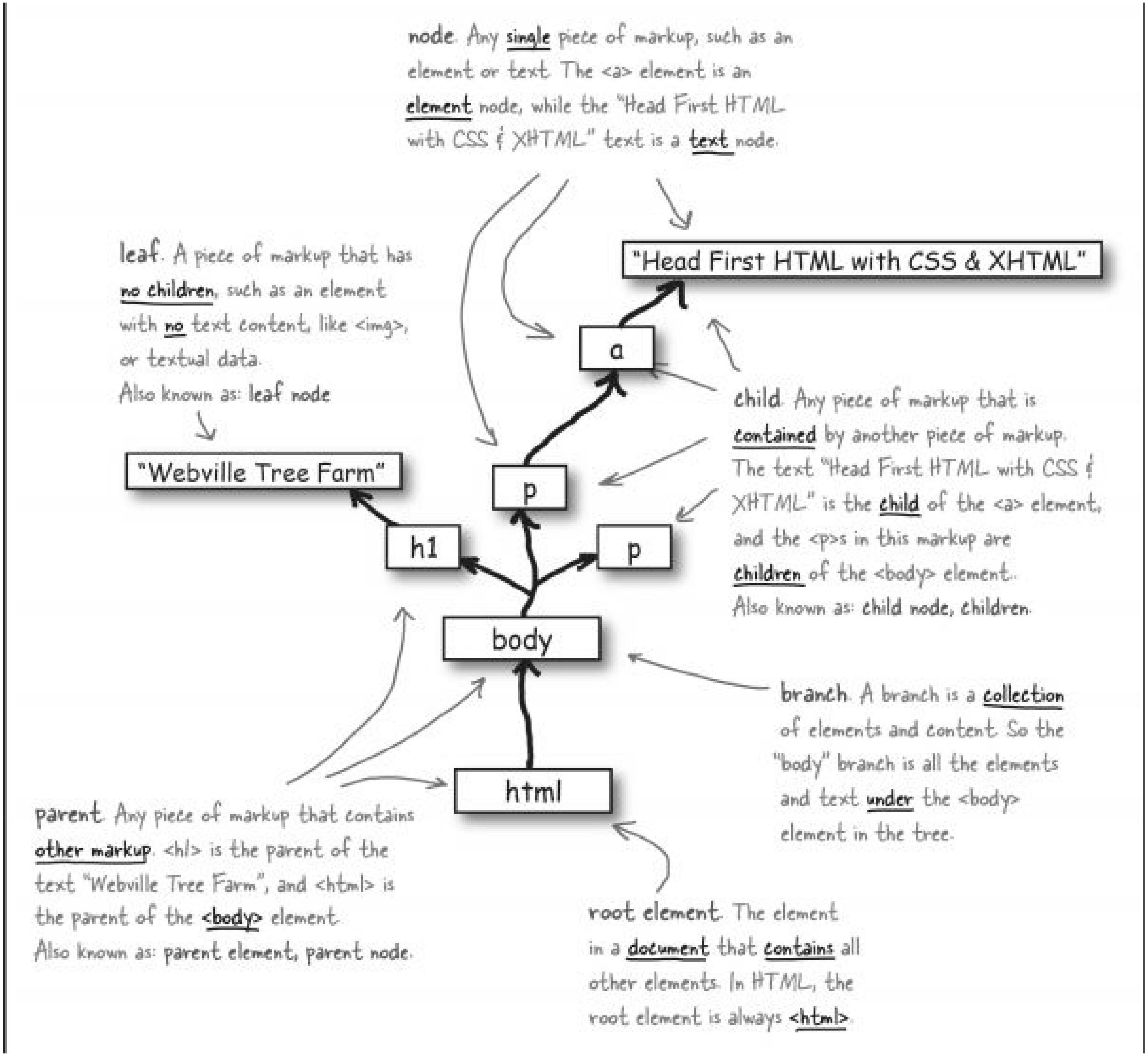
- getElementByID(“id”)
  - attach ID-attributes to HTML tags and access page elements by this notation, instead of having to wade through the hierarchy.
- appendChild()
  - Use appendChild() and .text or .innerHTML instead of += concatenation

DOM

- getElementByTagName(“div”)
  - Search through all the <id> tags.

DOM Exercise

DOM



node. Any **single** piece of markup, such as an element or text. The <a> element is an **element** node, while the "Head First HTML with CSS & XHTML" text is a **text** node.

leaf. A piece of markup that has **no children**, such as an element with **no** text content, like <img>, or textual data.
Also known as: **leaf node**

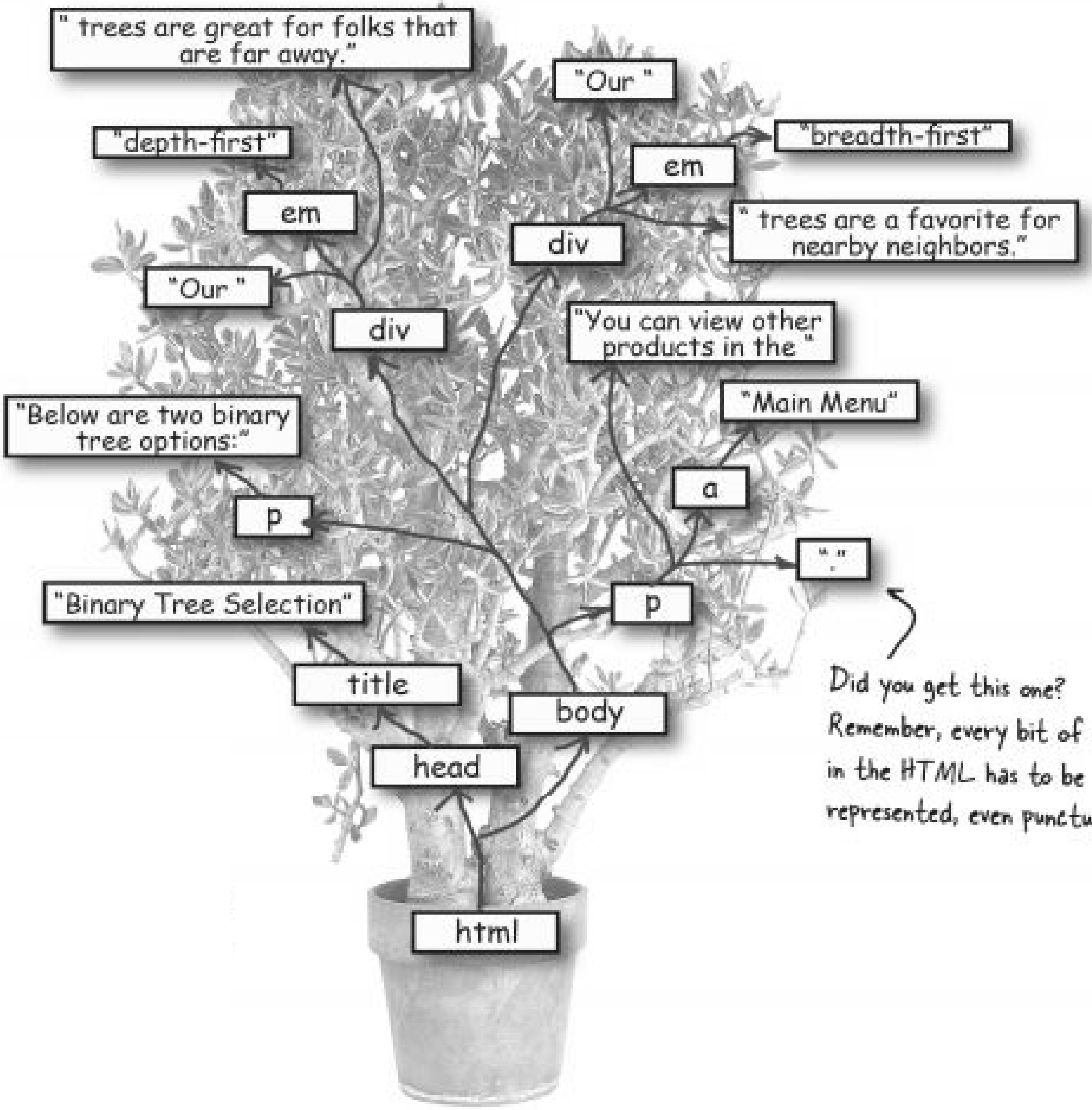
child. Any piece of markup that is **contained** by another piece of markup. The text "Head First HTML with CSS & XHTML" is the **child** of the <a> element, and the <p>s in this markup are **children** of the <body> element.
Also known as: **child node, children**.

branch. A branch is a **collection** of elements and content. So the "body" branch is all the elements and text **under** the <body> element in the tree.

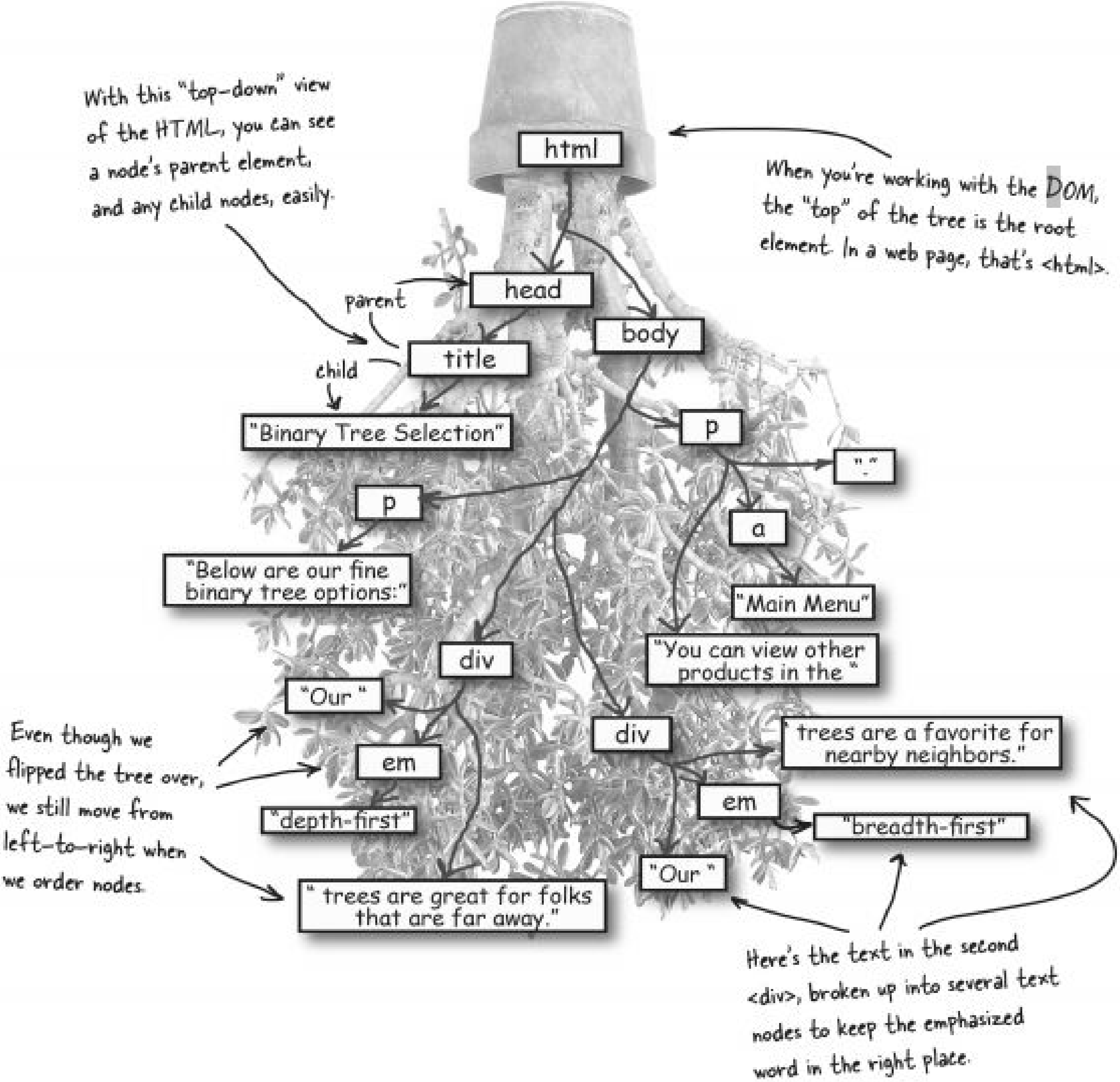
parent. Any piece of markup that contains **other markup**. <h1> is the parent of the text "Webville Tree Farm", and <html> is the parent of the **<body>** element.
Also known as: **parent element, parent node**.

root element. The element in a **document** that **contains** all other elements. In HTML, the root element is always **<html>**.

DOM Exercise 2


" trees are great for folks that are far away."
"Our "
"breadth-first"
"depth-first"
em
em
div
" trees are a favorite for nearby neighbors."
"Our "
div
"You can view other products in the "
"Main Menu"
"Below are two binary tree options:"
a
p
"."
"Binary Tree Selection"
p
title
body
head
html

Did you get this one? Remember, every bit of text in the HTML has to be represented, even punctuation.


html
head
body
title
"Binary Tree Selection"
p
"Below are our fine binary tree options:"
p
"."
a
"Main Menu"
"You can view other products in the "
div
"Our "
em
"depth-first"
" trees are great for folks that are far away."
div
" trees are a favorite for nearby neighbors."
em
"breadth-first"
"Our "
parent
child

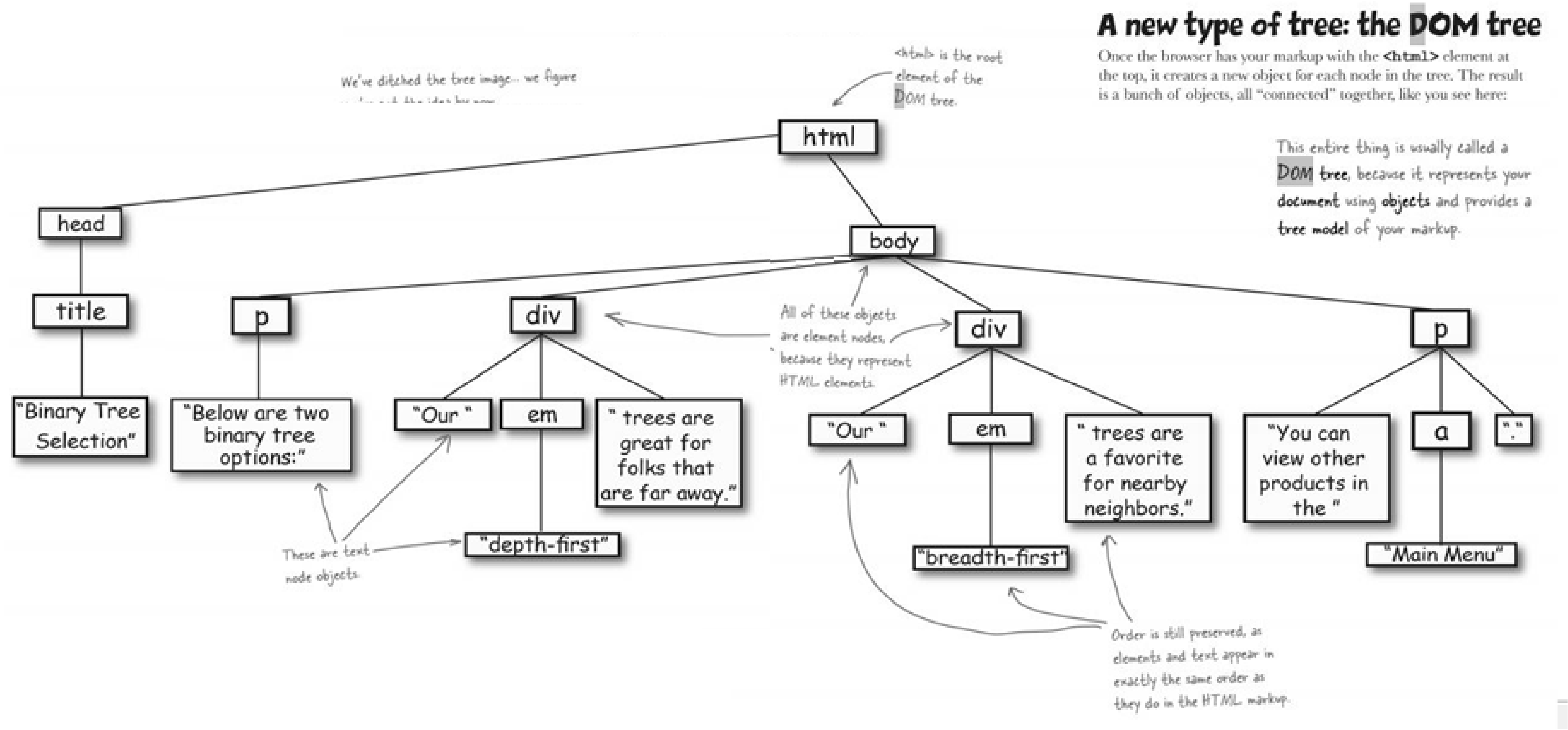
With this "top-down" view of the HTML, you can see a node's parent element, and any child nodes, easily.

When you're working with the DOM, the "top" of the tree is the root element. In a web page, that's <html>.

Even though we flipped the tree over, we still move from left-to-right when we order nodes.

Here's the text in the second <div>, broken up into several text nodes to keep the emphasized word in the right place.

The DOM Tree



A new type of tree: the DOM tree

Once the browser has your markup with the **<html>** element at the top, it creates a new object for each node in the tree. The result is a bunch of objects, all "connected" together, like you see here:

We've ditched the tree image... we figure

<html> is the root element of the DOM tree.

This entire thing is usually called a **DOM tree**, because it represents your **document** using **objects** and provides a **tree model** of your markup.

All of these objects are element nodes, because they represent HTML elements.

These are text node objects.

Order is still preserved, as elements and text appear in exactly the same order as they do in the HTML markup.

html
head
body
title
"Binary Tree Selection"
p
"Below are two binary tree options:"
div
"Our "
em
"depth-first"
" trees are great for folks that are far away."
div
"Our "
em
"breadth-first"
" trees are a favorite for nearby neighbors."
p
"You can view other products in the "
a
"Main Menu"
"."