What is the DOM?

The *Document Object Model*, abbreviated DOM, is a powerful tree-like structure that allows programmers to conceptualize hierarchy and access the elements on a web page.

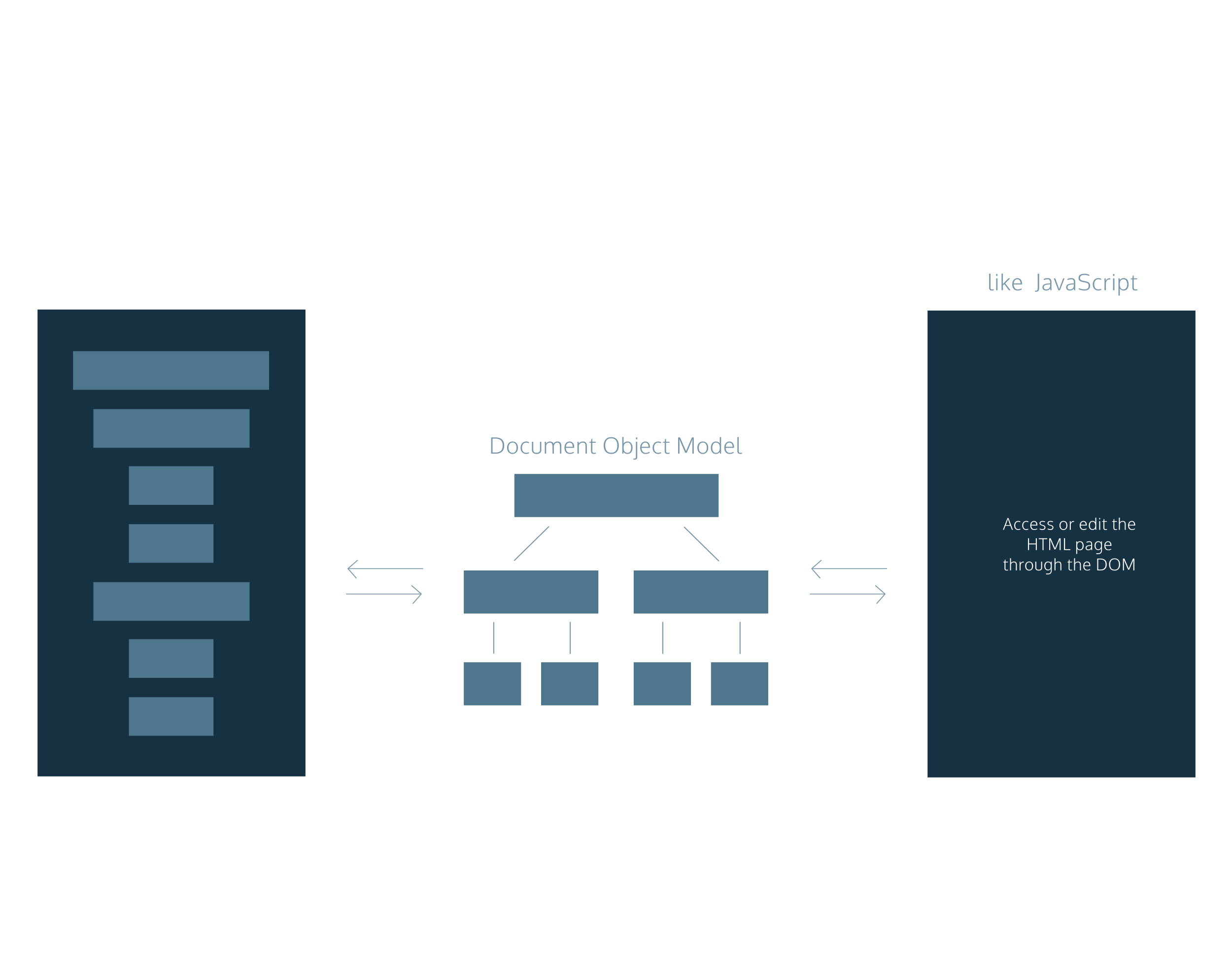
The DOM is one of the better-named acronyms in the field of Web Development. In fact, a useful way to understand what DOM does is by breaking down the acronym but out of order:

* The *DOM* is a logical tree-like **M**odel that organizes a web page’s HTML **D**ocument as an **O**bject.

Note: There are other types of documents, such as XML and SVG, that are also modeled as DOM structures.

The DOM is a language-agnostic structure implemented by browsers to allow for web scripting languages, like JavaScript, to access, modify, and update the structure of an HTML web page in an organized way.

For this reason, we like to think of the DOM as the link between an HTML web page and scripting languages.



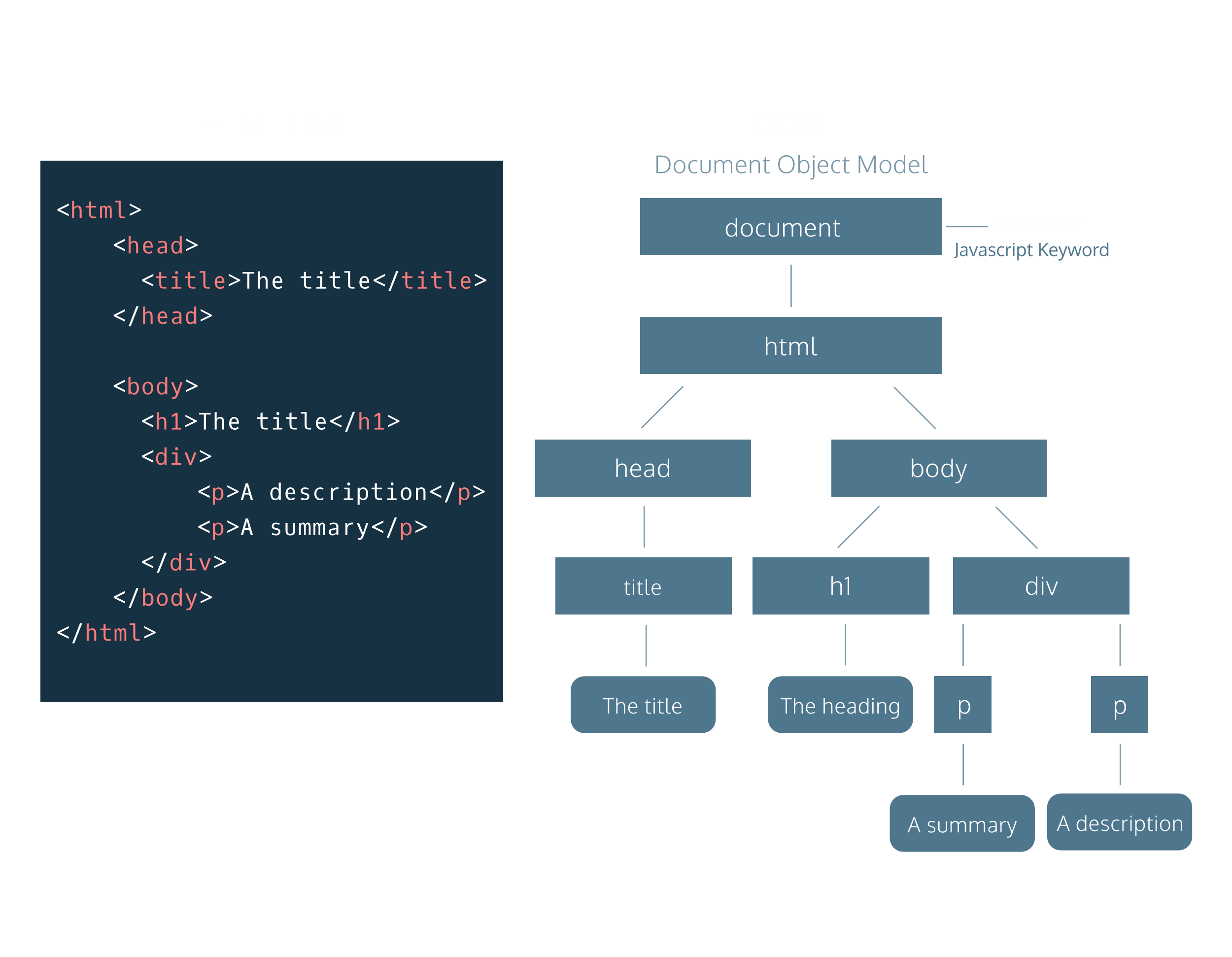
The DOM as a Tree Structure

Tree-like modeling is used in many fields, including evolutionary science and data analytics. Perhaps you’re already familiar with the concept of family trees: these charts represent the familial relationships amongst the descendants of a given family name.

The DOM tree follows similar logic to that of a family tree. A family tree is made up of family members and their relationships to the family name. In computer science, we would call each family member a node.

We define a *node* as an intersecting point in a tree that contains data.

In the DOM tree, the top-most node is called the *root node*, and it represents the HTML document. The descendants of the root node are the HTML tags in the document, starting with the <html> tag followed by the <head> and <body> tags and so on.



Parent Child Relationships in the DOM

Following the metaphor of a family tree, let’s define some key terminology in the DOM hierarchy:

A *parent node* is the closest connected node to another node in the direction towards the root.

A *child node* is the closest connected node to another node in the direction away from the root.

Knowing these terms will allow you to understand and discuss the DOM as a tree-like structure. In fact, you will also see this terminology used when referring to the nesting structure of HTML code. Programmers refer to elements nested inside other elements as the children elements and parent elements respectively.

HTML

<!DOCTYPE html>

<html>

  <head>

    <title>The title</title>

  </head>

  <body>

    <h1>The heading</h1>

    <div>

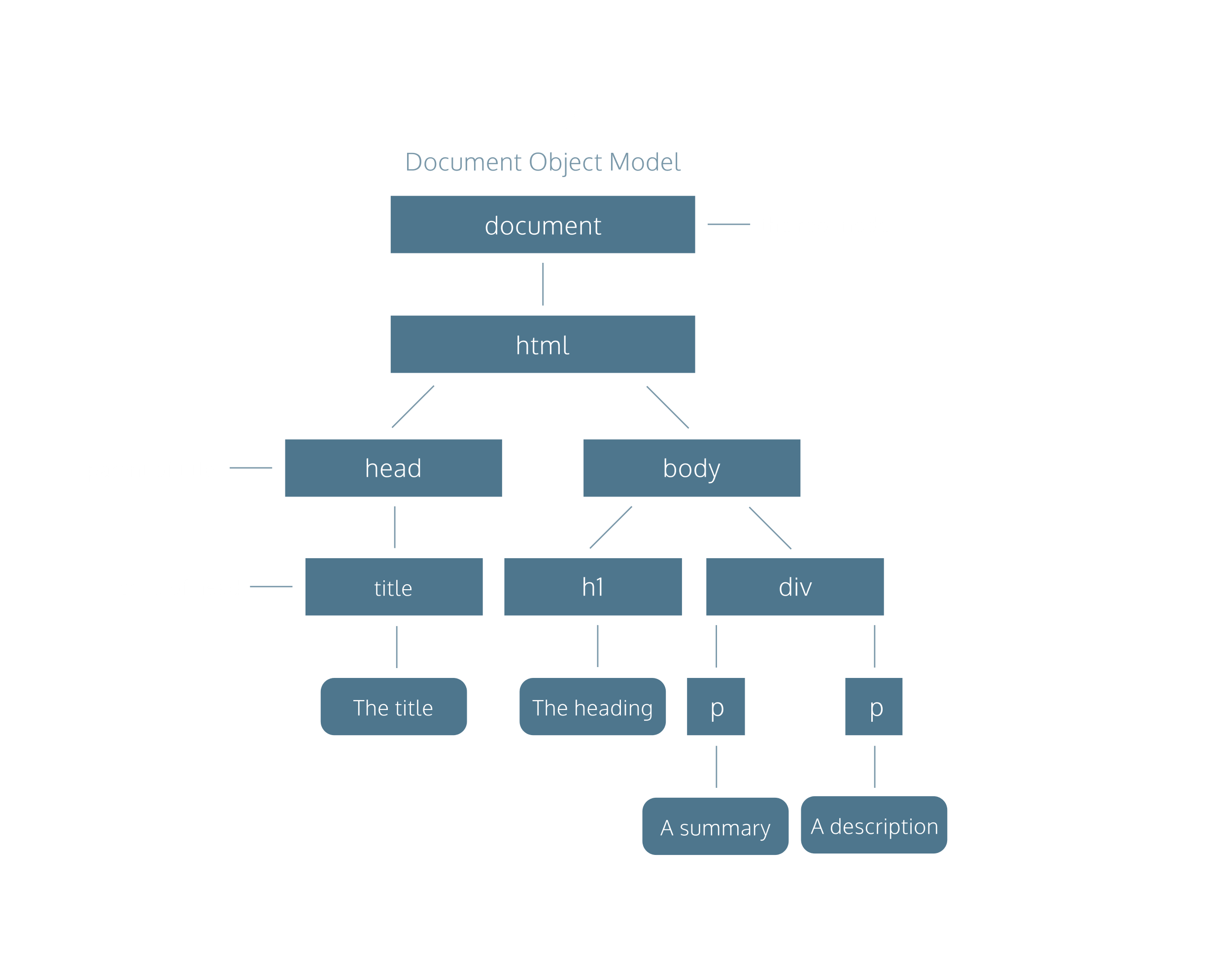
      <p>A summary</p>

      <p>A description</p>

    </div>

  </body>

</html>

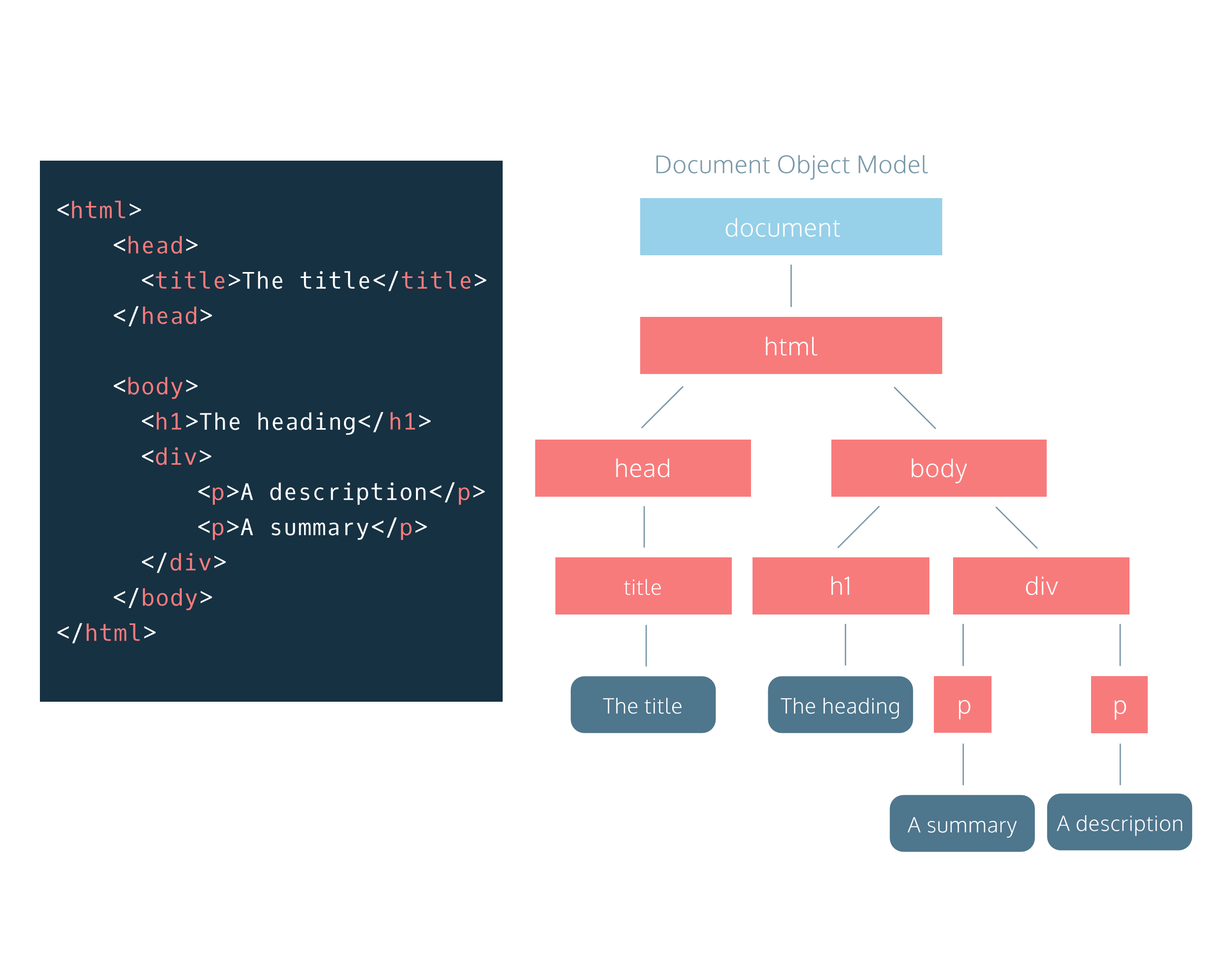


Nodes and Elements in the DOM

As mentioned, a node is the equivalent of each family member in a family tree. A node is an intersecting point in a tree that also contains data.

There are nine different types of node objects in the DOM tree. In our diagram, the node objects with the sharp-edge rectangles are of the type [*Element*](https://developer.mozilla.org/en-US/docs/Web/API/Element), while the rounded edge rectangles are of type [*Text*](https://developer.mozilla.org/en-US/docs/Web/API/Text), because they represent the text inside the HTML paragraph elements.

When trying to modify a web page, the script will mostly interact with the DOM nodes of type element. Elements are the building units of HTML web pages, they contain everything between an opening tag and a closing tag. If the tag is a self-closing tag, then that is the element itself.



Attributes of Element Node

DOM element nodes model elements in an HTML document.

Much like an element in an HTML page, the DOM allows us to access a node’s attributes, such as its class, id, and inline style.

In the diagram to the right, we have highlighted the paragraph element with an id of “bio” in the HTML document. If we were accessing that element node in our script, the DOM would allow us to tweak each of those attributes, or simply access them to check their value in the code.

