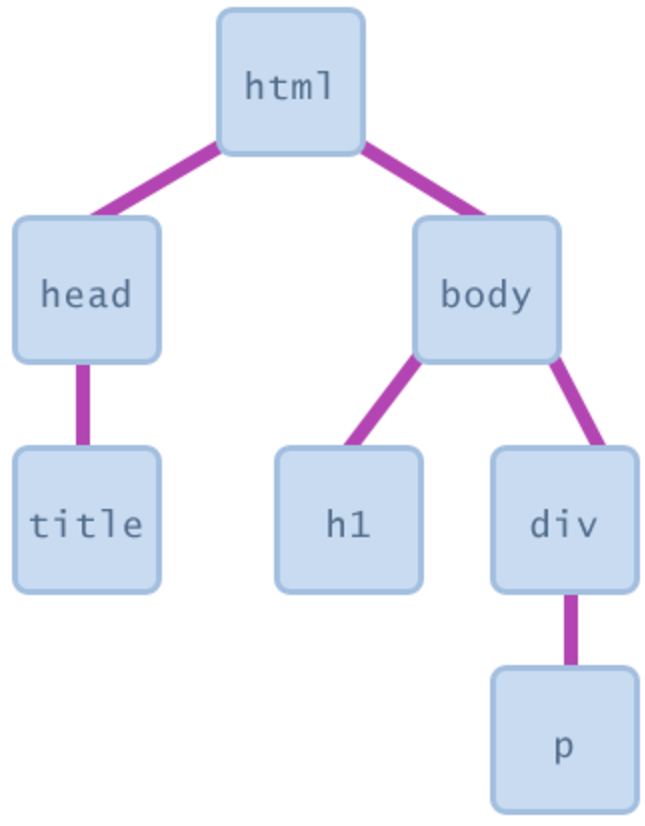
As you learn about HTML, you are bound to come across the acronym DOM. Since the DOM is the foundation of almost everything JavaScript does for us on the web, it’s essential for you to understand how to get around within it. In this section, we will:

-Explain what the DOM is

-Why it is important to understand

- How to use Chrome’s Developer tools to interact with the DOM

The **DOM**, or the **Document Object Model**, is the internal , programmatic, representation of a web page. You can kind of think of it like a virtual map, or a model, of the webpage. It is a data structure that describes the hierarchy of all the elements (sometimes referred to as **nodes**) on a page. These elements understand their relationship to other elements around them, and contain a ton of information about themselves.



Above, on the left, you see the visual representation of a sample webpage’s DOM. On the right, you see that same webpage’s actual HTML code. On close inspection of the picture on the left, you’ll see that the this representation of the DOM kinda looks like an upside down tree. Because of this, you will often hear this described as the **DOM tree**.

In fact, the words we use to describe the relationships between nodes make the DOM sound like a family tree. For example, the html element above has two **children elements**, head and body. These elements can be considered **siblings.** Head and body elements also have a single **Parent** element, the html element.  
  
  
At the base structure of every DOM Tree is the document master container. (not pictured) It is referenced as **document.** This refers to the entire document, and all elements within it. Since this document serves as an entry point into the web page's content, developers like us can interact with and Manipulate the items within it.

With this course, we will use the Chrome Developer tools that come with Google Chrome to interact/manipulate the dom. If you don’t have Chrome yet, please take a minute to download it at <https://www.google.com/chrome/browser/>.

So before we get started working with the DOM, let’s learn a bit about these new amazing tools!

So, what is Chrome Dev Tools? It is a set of web authoring and debugging tools built into Google Chrome. It provides web developers deep access into the internals of the browser and their web application. There are two ways you can open Chrome Dev tools. First, for either, make sure you have your Chrome Browser open, then you can either:

1. Select the **Chrome menu** at the top-right of your browser window, then select **Tools** > **Developer Tools**.
2. Right-click on any page element and select **Inspect Element**.

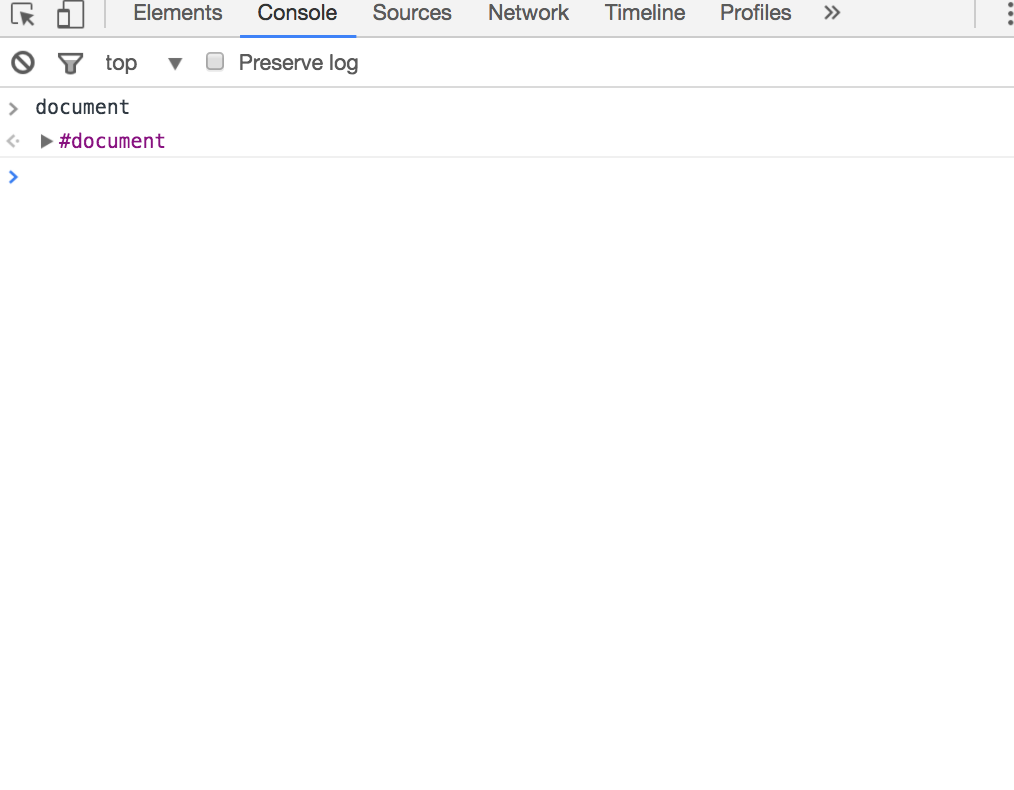
When you open Chrome Dev tools, you’ll notice that it has a couple of panels that you can pick from. These Panels include:

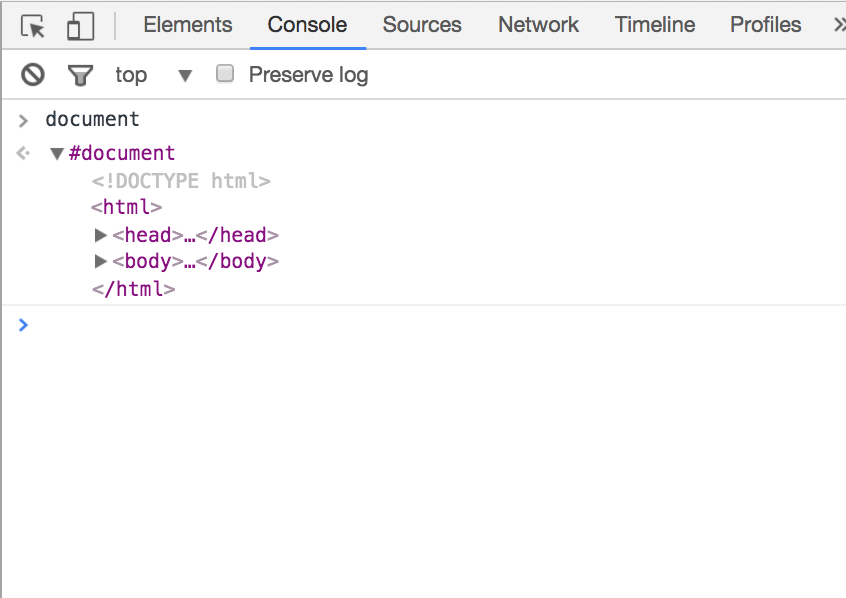
* **Element Panel**- Shows you all the HTML elements on the page.
* **Console Panel**-Allows us us to interact with Javascript and Debug our current webpage.
* **Sources Panel**- Allows us to put breakpoints in our code and actually step through our code, line-by-line.
* **Network Panel**- tells us the sequence and load time of the webpage
* **Timelines Panel**- Used to measure refresh rates, memory usage, and the time it takes for events to execute.
* **Profiles Panel**- Analyzes CPU usage. This panel, and the Timelines panel, are useful in understanding *why* a page loads slowly.
* **Application**- Gives us the ability to inspect and manage storage, databases, and caches
* **Security Panel -** Tells us whether the current web page is compliant with 2-3 basic security checks.
* **Audits Panel**- Runs a few common tests on our webpage (network usage, memory consumption) and gives us suggestions on making improvements to our webpage.

For this section, we will mostly use the **Console** **Panel.**  We will use this panel, along with some simple Javascript commands to interact with the Dom , **Traverse** (Move from one element to another) in the Dom, and manipulate it to change and show different things! We will use a sample HTML file to start working with the Dom. Please download the file from this Location:

<https://www.dropbox.com/s/5qi6eqsocwyo94p/example.html?dl=0>

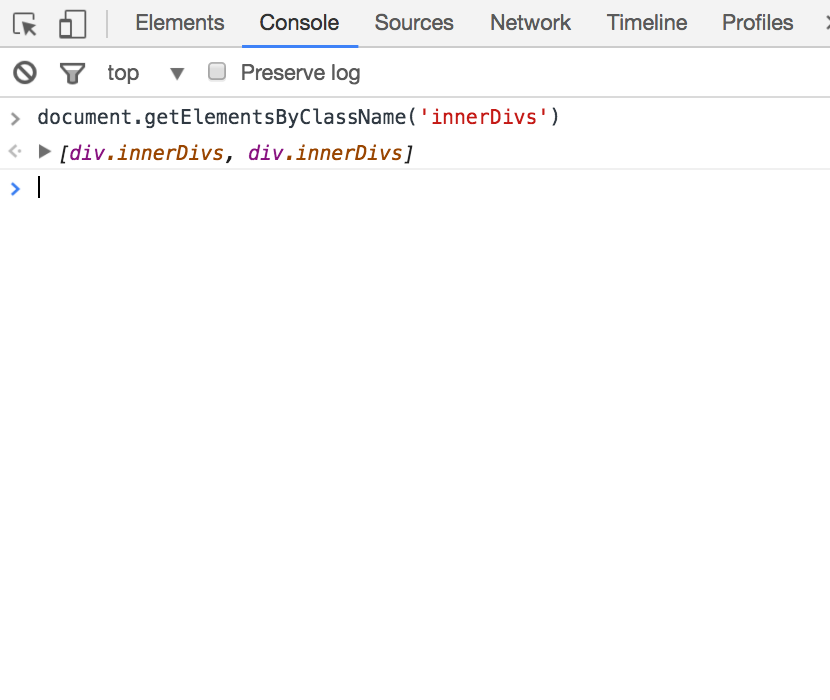
Inside of your Console type document and hit enter. This should bring up a screen that looks like this:



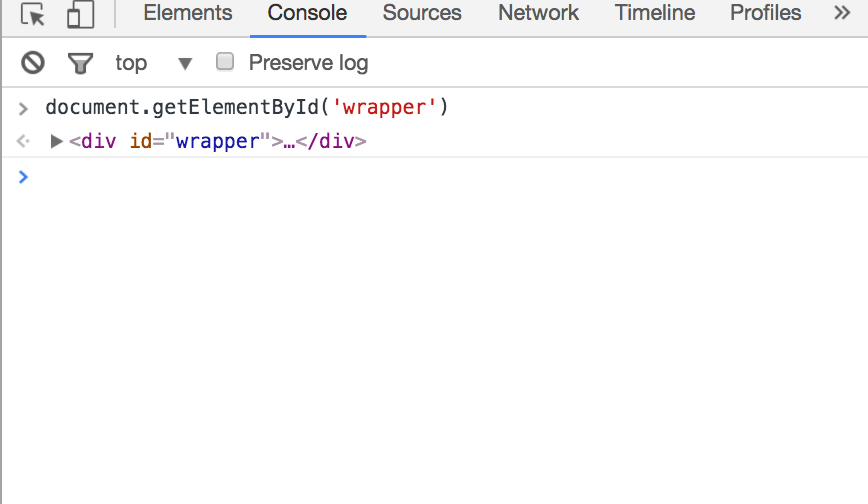
You should have noticed that when you hit enter, it brought up a purple #document, with a right arrow to the left of it. If you click on the arrow, you should be presented with the following:

This shows all the nodes that are inside of the document. We can also click on the arrows inside of the document to reveal what that container is holding.

We can use the .getElementsByClass method to get all of the elements with the same class name that we pass it , and puts the results inside of an array:

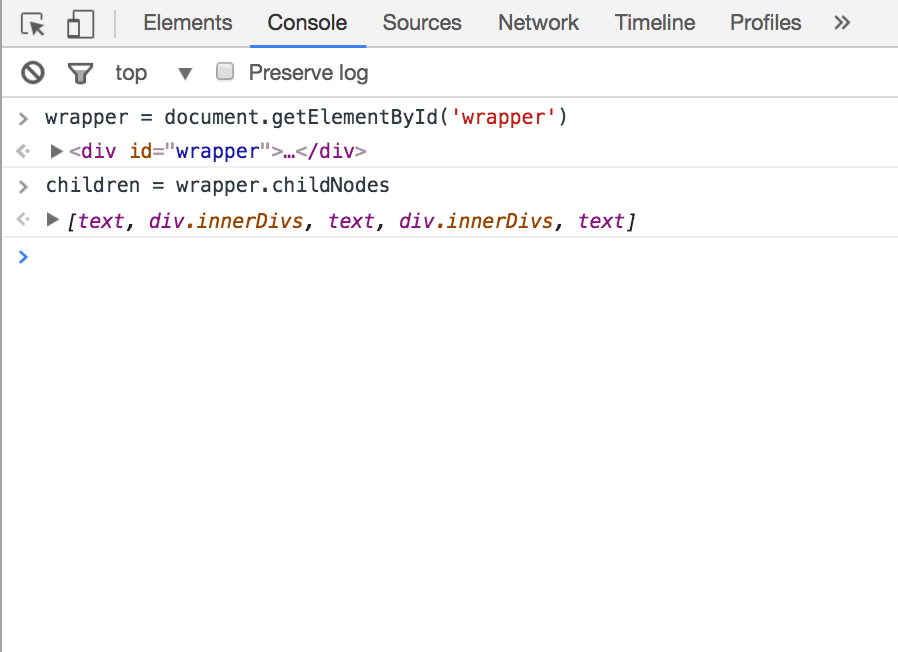


Likewise, we can also use the .getElementById property to get the matching element. Notice that this one is singular, and will only return that one element.



So far, we have mainly used document as our root to access other elements on our page. However this doesn’t have to be our starting point every time we want to access an element or its contents. Since we have access to Javascript, we can use variables to store what returns from the methods we use.

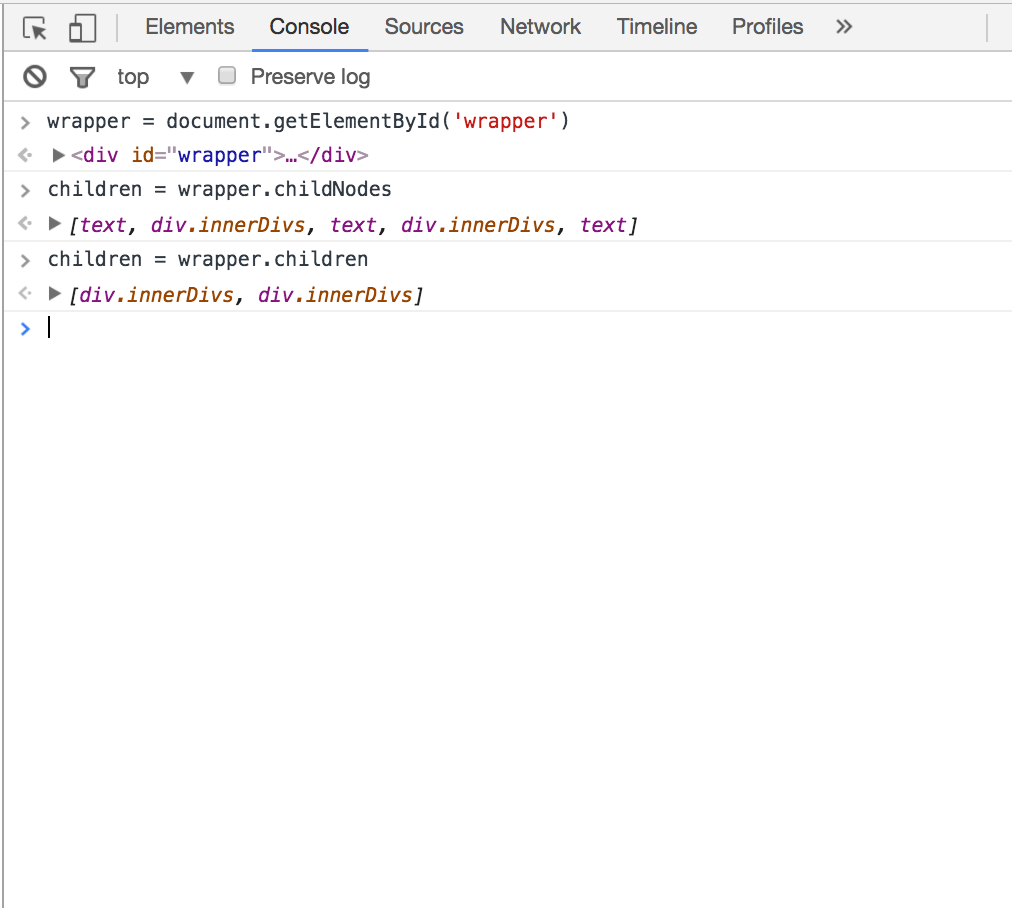
Let’s see this in action. Lets use the getElementById(‘wrapper’) method and store the results in a variable called wrapper. After this. Lets call the .childNodes method on the variable and store the results of that in a variable called children. Your code should look something like this:



Now, at anytime in your console (given that you don’t refresh the page) you can access the children of wrapper simply by typing children in it.

Notice we now have array-like container that contains both divs that were inside of the wrapper element. This container is not actually an array, but rather a HTML **collection** – a special array-like iterable object. Keep in mind that we can still use some of the common methods (like .length or indexing with bracket notation ) we use with arrays on these collections.

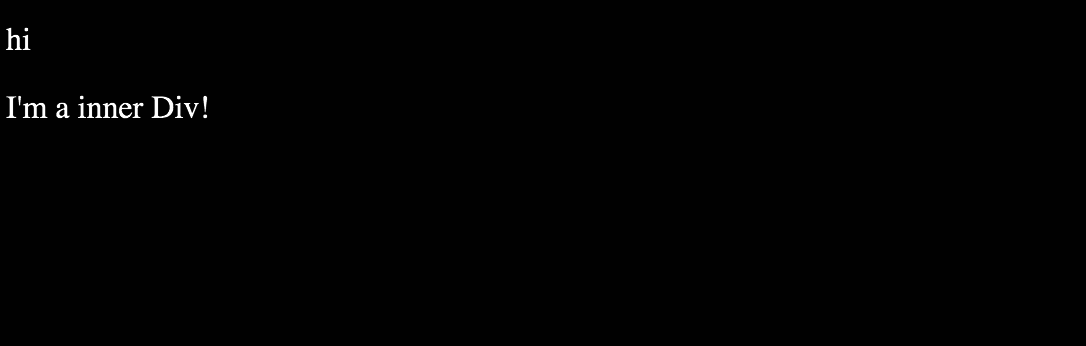
We also have a couple elements called text inside this array which are nothing more than us having white space between the elements our html file, or us using the return key to put elements on different lines. If this is undesirable, we can use the children method instead of the childNodes method. This will return an array with just the actual html elements inside of it .



So far, we have been using methods to traverse through the dom tree. This is great, but what if we wanted to change what a element contains?We could use use .innerHTML on the element we want to contain different data. Let’s try that on the first div In our children variable:



As you see in the above diagram, I went into the children array, and grabbed the element at the 0th index, and changed it to contain ‘<p>hi</>. Upon hitting enter, you should see your web page change to show your updates to the document.



You now have the basics down to being able to traverse the DOM with Javascript. However, there are a ton of other built-in Methods you could use to help get your task done! Here is a list of a few others you might want to look up:

.firstChild

.lastChild

.count

.appendChild

.className

.getElementsByTagName

.nextSibling

Later on, you will be using other libraries such as React and Jquery to help you manipulate the DOM. However, it is important for you to know what is happening behind the scenes!