

Introduction to the Document Object Model

You're About to learn

- ◉ What is the DOM?
- ◉ Why should we care?
- ◉ DOM Manipulation
 - Searching the DOM
 - How to traverse the DOM
 - How to change the DOM

What is the DOM?



The Document Object Model is what allows web pages to render, respond to user events and change

HTML vs DOM

```
<body>

<h1>Hello</h1>

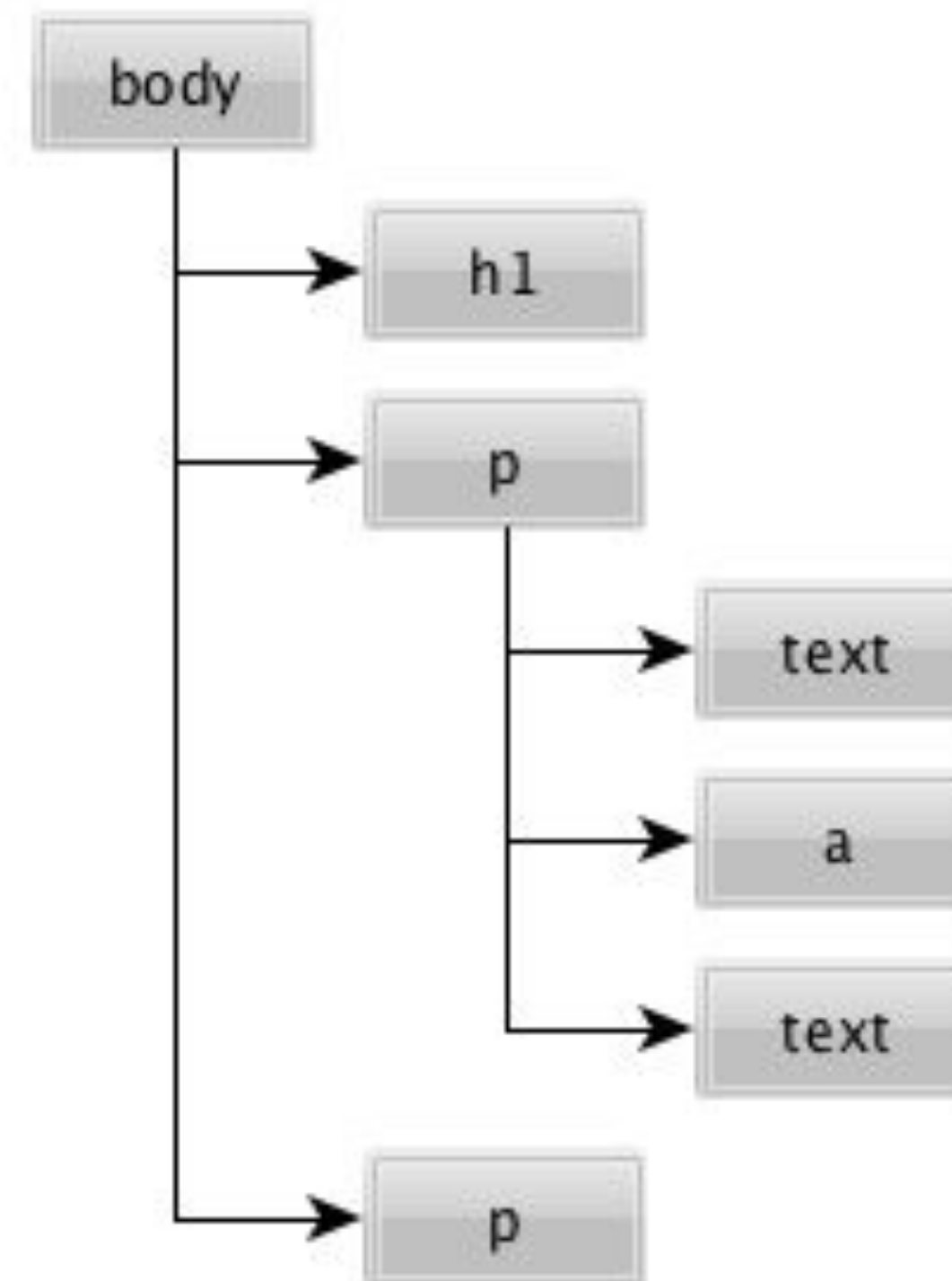
<p>

  Check out my

  <a href="/page">Page!</a>

  It's the best page out there
</p>

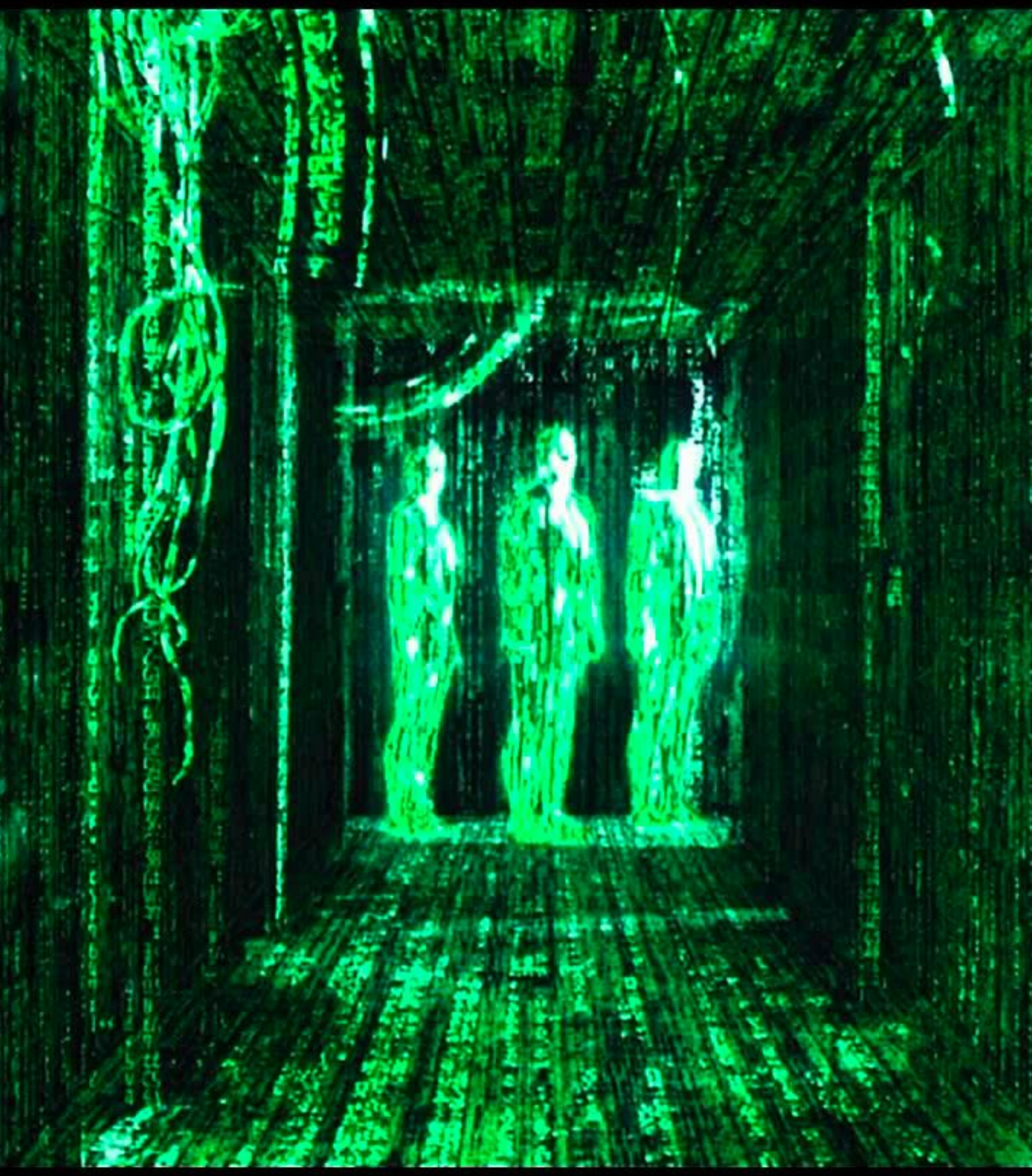
  <p>Come back soon!</p>
</body>
```





1. The first part of the document is a list of names and their corresponding dates. The names are written in a cursive script, and the dates are in a standard font. The list is organized into two columns, with names on the left and dates on the right. The names are: John Doe, Jane Smith, and Robert Johnson. The dates are: 1990, 1991, and 1992. The list is as follows:

Name	Date
John Doe	1990
Jane Smith	1991
Robert Johnson	1992

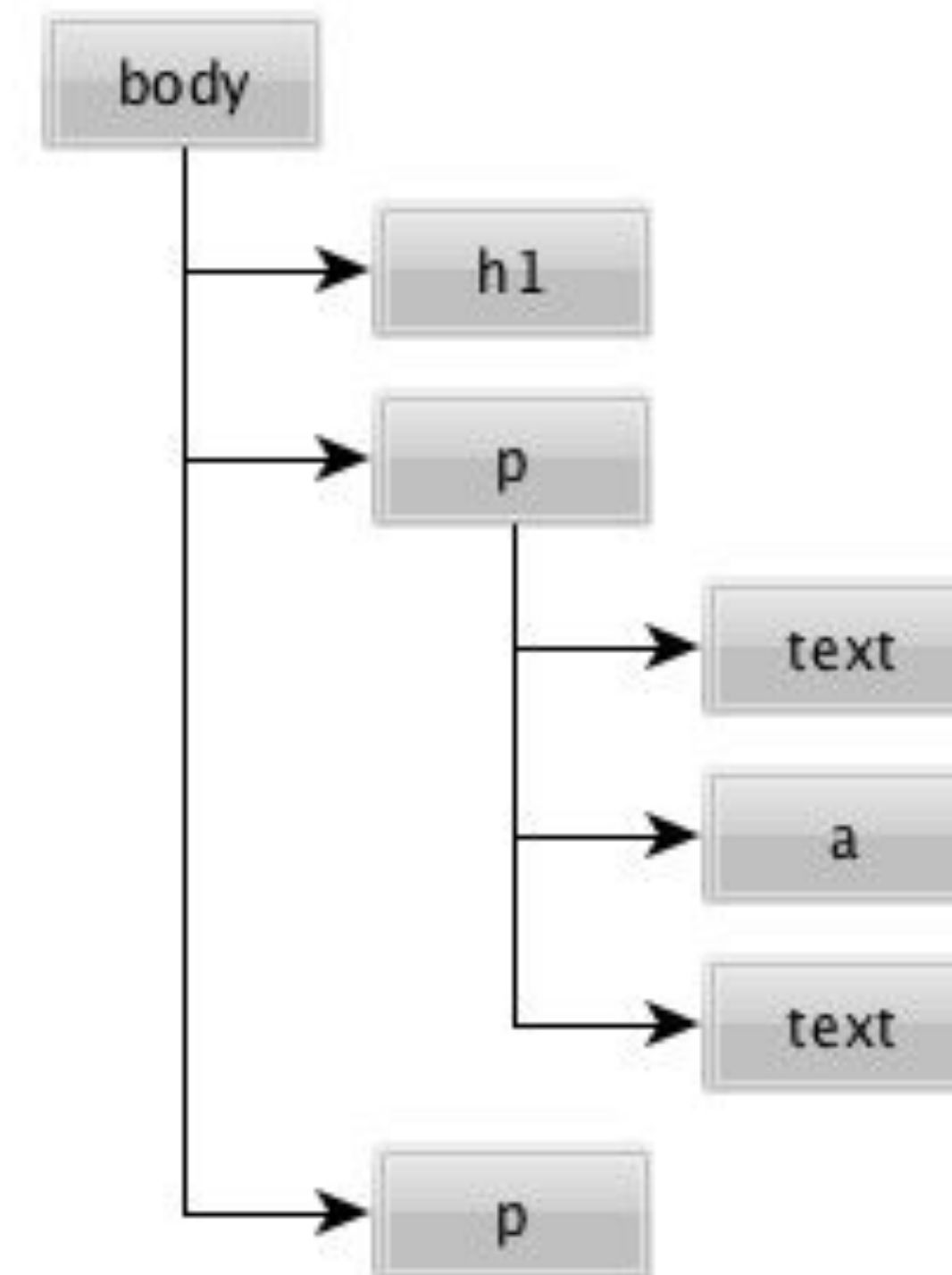


2. The second part of the document is a list of names and their corresponding dates. The names are written in a cursive script, and the dates are in a standard font. The list is organized into two columns, with names on the left and dates on the right. The names are: John Doe, Jane Smith, and Robert Johnson. The dates are: 1990, 1991, and 1992. The list is as follows:

Name	Date
John Doe	1990
Jane Smith	1991
Robert Johnson	1992

The DOM is a Tree

- ◉ Trees are a data structure from computer science
- ◉ The main idea here: There is a Node that branches into other Nodes (its children Nodes)
 - ◉ Each Node can have 0 to many children Nodes
 - ◉ Nodes can have 0 or 1 parent
 - ◉ Nodes can have 0 to many Sibling Nodes



Developer Tools

The screenshot shows a web browser window with several tabs: "DOM Lecture Outline - draw", "JS Bin - Collaborative Java", "Google Analytics", and "Issues · FullstackAcademy". The address bar shows the URL "https://github.com/FullstackAcademy/newlearn/issues?utf8=✓&q=is%3Aissue+is%3Aopen+comments". The page displays the GitHub repository interface for "FullstackAcademy / newlearn" (PRIVATE), with 24 issues, 11 stars, and 5 forks. The "Issues" tab is selected, and a search filter "is:issue is:open comments" is applied. The Chrome DevTools interface is open at the bottom, showing the "Elements" panel on the left and the "Styles" panel on the right. The "Elements" panel shows the HTML structure of the page, with the "body" element selected. The "Styles" panel shows the default styles for the "body" element, including "display: block" and "margin: 8px".

Elements

```
<!DOCTYPE html>
<html lang="en" class=" is-copy-enabled">
  <head prefix="og: http://ogp.me/ns# fb: http://ogp.me/ns/fb# object: http://ogp.me/ns/object# article: http://ogp.me/ns/article# profile: http://ogp.me/ns/profile#">...</head>
  <body class="logged_in env-production macintosh vis-private">
    <a href="#start-of-content" tabindex="1" class="accessibility-aid js-skip-to-content">Skip to content</a>
    <div class="header header-logged-in true" role="banner">...</div>
    <div id="start-of-content" class="accessibility-aid"></div>
    <div id="js-flash-container">
    </div>
    <div role="main" class="main-content">...</div>
    <div class="container">...</div>
    <div id="ajax-error-message" class="flash flash-error">...</div>
    <script crossorigin="anonymous" integrity="sha256-Ln/D0mSiC0E4PehbgVN5vsz/VsH5d3FFFdTKx4I07z4=" src="https://assets-cdn.github.com/assets/frameworks-2e7fc3d264a208e1383de85b815379beccff56c1f977714515d4cac7820eef3e.js"></script>
    <script async="async" crossorigin="anonymous" integrity="sha256-0ZauYVQU9+hAsJTbTqtuSyWQK5rrZ0UHUHJlozDGwdA=" src="https://assets-cdn.github.com/assets/github-d196ae615414f7e840b094db4eab6e4b25902b9aeb674507521265a330c6c1d0.js"></script>
    <div class="js-stale-session-flash stale-session-flash flash flash-warn flash-banner hidden">...</div>
    <div class="facebox" id="facebox" style="display:none;">...</div>
    <script id="hiddenSubmitdiv" style="display: none;"></script>
    <script>...</script>
  </body>
</html>
```

Styles

```
Filter
element.style {
}

media="all"
body {
  word-wrap: break-word;
}

media="all"
body {
  font: 13px/1.4 Helvetica, arial, nimbussansl, liberationans, freesans, clean, sans-serif, "Segoe UI Emoji", "Segoe UI Symbol";
  color: #333;
  background-color: #fff;
}

media="all"
body {
  margin: 0;
}

media="all"
* {
  box-sizing: border-box;
}

body {
  display: block;
  margin: 8px;
}

Inherited from html.is-copy-enabled
media="all"
html {
  font-family: sans-serif;
```


Why care?

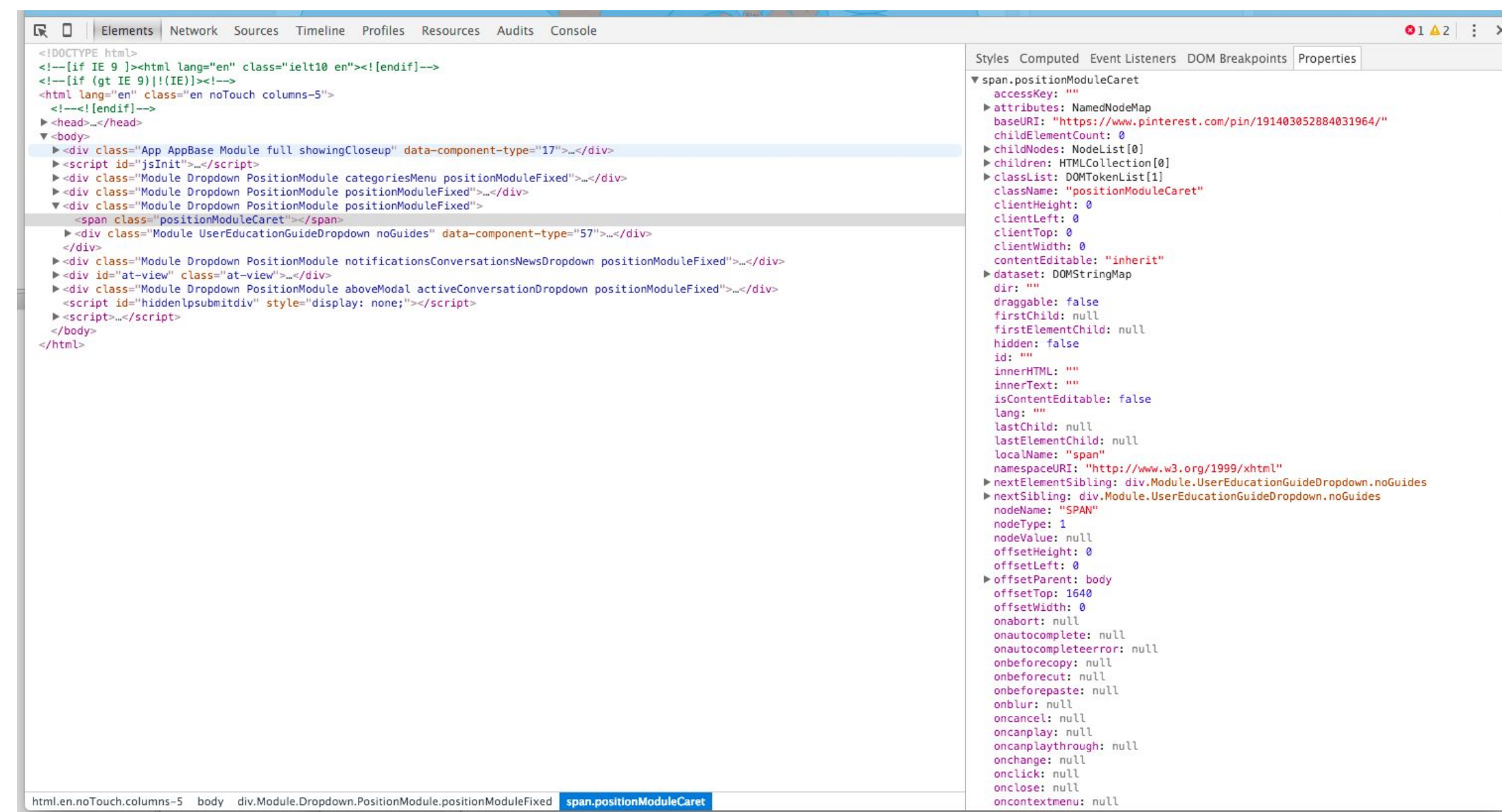


The DOM makes possible to use JavaScript to manipulate the document content and structure

Nodes have lots of Attributes

- ◉ Nodes are JavaScript Objects
- ◉ Nodes have Attributes that are JavaScript properties
- ◉ Attributes define how the Node looks and responds to User activity

ONE NODE



Hundreds of properties!

The *document* Object

- ◉ Global reference to the DOM entry point
- ◉ Provides methods for:
 - Navigating the DOM
 - Manipulating the DOM
- ◉ The ***document*** object is the important connection between the DOM and JavaScript code

Searching the DOM



Searching the DOM

- ◉ `getElementById` (find nodes with a certain ID attribute)
 - `document.getElementById("will");`
- ◉ `getElementsByClassName` (find nodes with a certain CLASS ATTRIBUTE)
 - `document.getElementsByClassName("will");`
- ◉ `getElementsByTagName` (find nodes with a certain HTML tag)
 - `document.getElementsByTagName("div");`
- ◉ `querySelector, querySelectorAll` (search using CSS selectors)
 - `document.querySelector("#will.will:first-child");`

Array-Like Objects? Bleh!

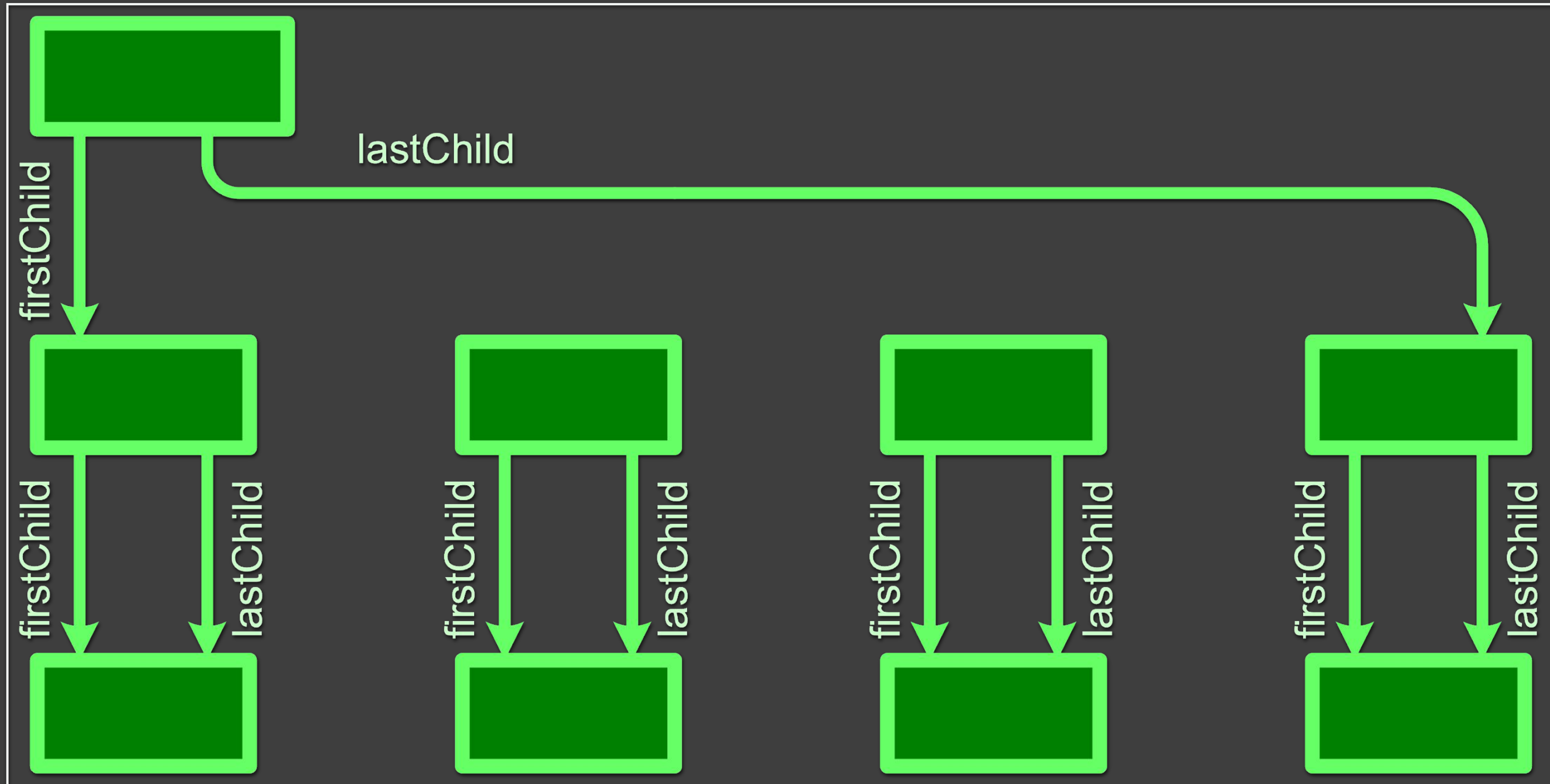
- ◉ `const realArr = [].prototype.slice.call(arrayLike)`
- ◉ `const realArr = Array.from(arrayLike)`
- ◉ `const realArr = [...arrayLike]`

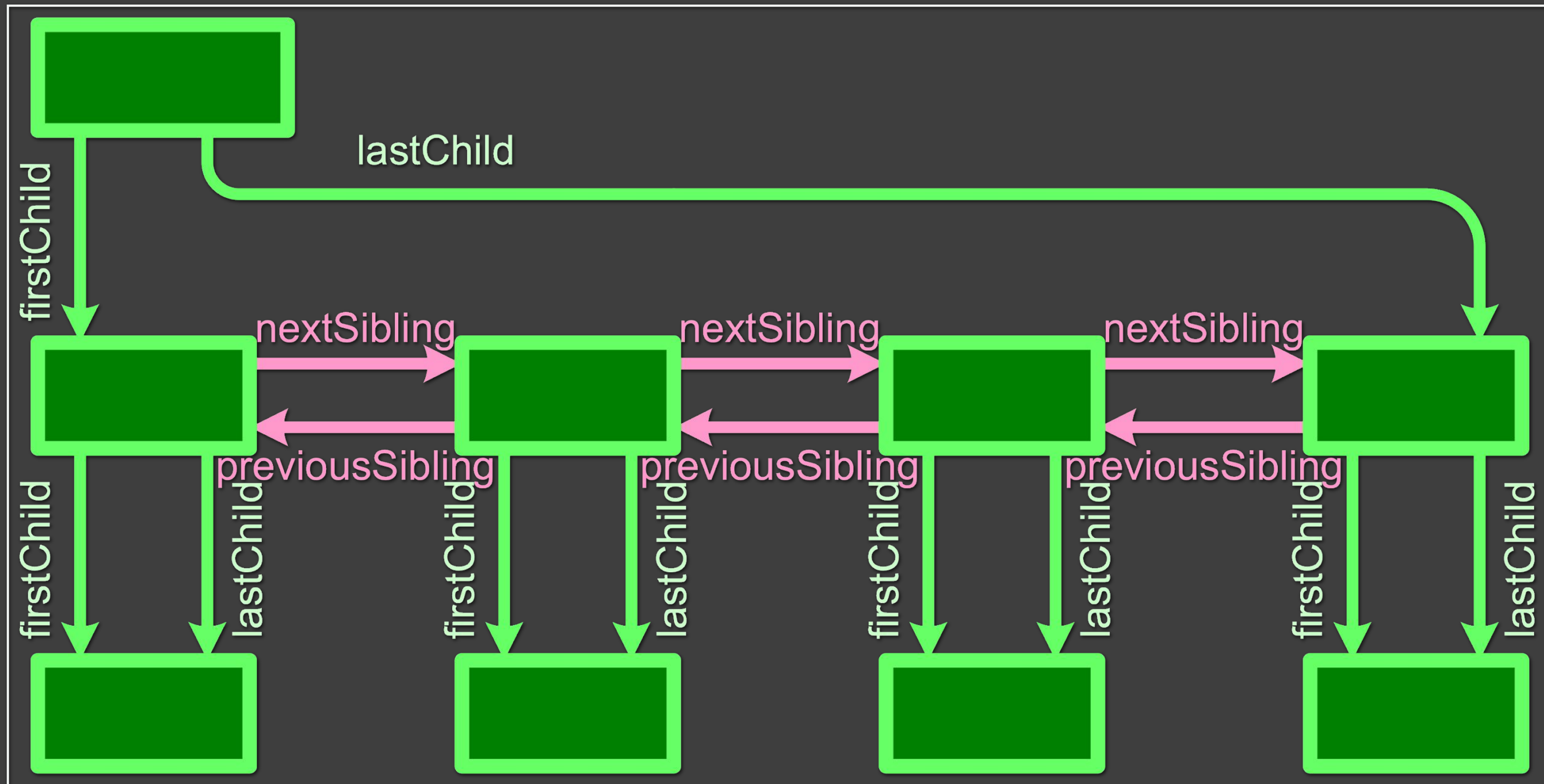
Traversing the DOM

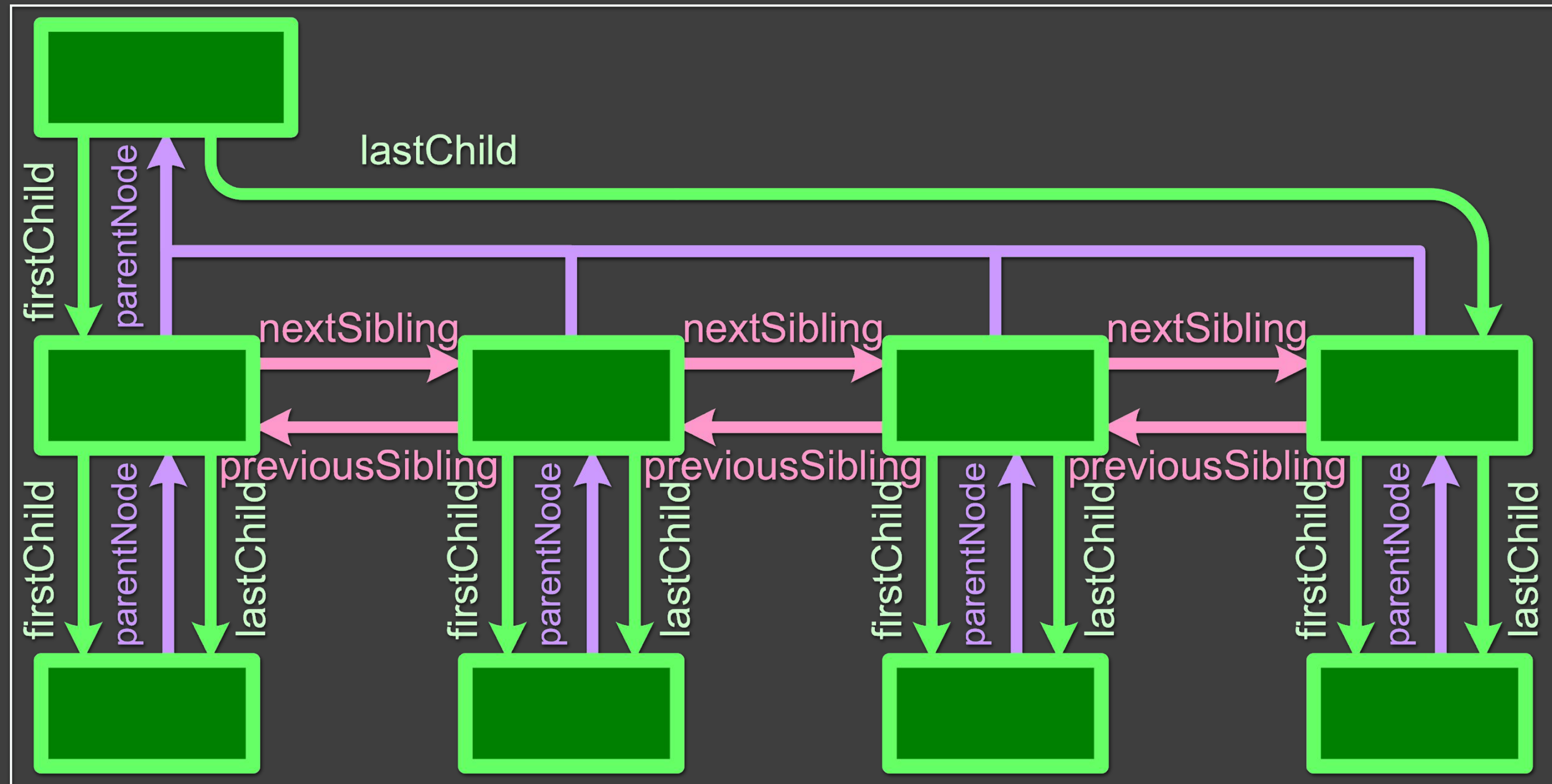


Traversing the DOM

- ◉ Tree Structures are easy to navigate:
 - At any point in the DOM you are at a Node
 - No matter where you go, you're still at a Node
 - Child
 - Parent
 - Sibling
 - All Nodes share similar DOM navigation methods







Traversing the DOM

- ◉ **Access children**

- `element.children`, `element.lastChild`, `element.firstChild`

- ◉ **Access siblings**

- `element.nextElementSibling`, `element.previousElementSibling`

- ◉ **Access parent**

- `element.parentElement`

Changing the DOM



Changing style attributes

```
element.style.backgroundColor = "blue";
```

◦ CSS

- background-color
- border-radius
- font-size
- list-style-type
- word-spacing
- z-index

◦ JavaScript

- backgroundColor
- borderRadius
- fontSize
- listStyleType
- wordSpacing
- zIndex

Changing CSS Classes

- ◉ ***className*** attribute is a string of all of a Node's classes
- ◉ ***classList*** is HTML5 way to modify which classes are on a Node

```
document.getElementById("MyElement").classList.add('class');
```

```
document.getElementById("MyElement").classList.remove('class');
```

```
if ( document.getElementById("MyElement").classList.contains('class') )
```

```
document.getElementById("MyElement").classList.toggle('class');
```

Creating Elements

- ◉ Create an element
 - ◉ `document.createElement(tagName)`
- ◉ Duplicate an existing node
 - ◉ `node.cloneNode()`
- ◉ Nodes are just free floating, not connected to the document itself, until you ***link*** them to the DOM.

Adding elements to the DOM

- ◉ Insert newNode at end of current node
 - ◉ `node.appendChild(newNode);`
- ◉ Insert newNode at end of current node
 - ◉ `node.prependChild(newNode);`
- ◉ Insert newNode before a certain childNode
 - ◉ `node.insertBefore(newNode, sibling);`

Removing Elements

- ◉ Removes the oldNode child.
 - ◉ `node.removeChild(oldNode) ;`
- ◉ Quick hack:
 - `oldNode.parentNode.removeChild(oldNode);`

LAB

