https://github.com/zgordon/wcdc-2017

Up & Running w Vanilla JavaScript

Zac Gordon @zgordon
javascriptforwp.com

HTML

Markup language using tags

CSS

Presentation language using key value pairs

JavaScript

An actual programming language

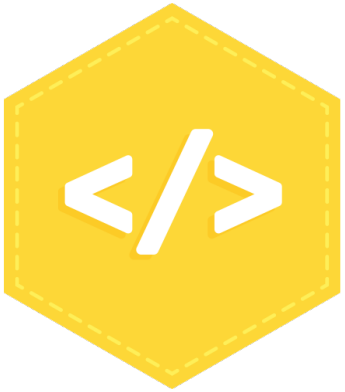
It's not that JavaScript is a hard or weird language. However, almost everything you do in JavaScript involves an API. That makes things more complicated.

Web APIs

Standards made available to allow JavaScript (ECMAScript) to interact with the browser (and more).

W3C and WHATWG standards bodies.

JavaScript & APIs Made Simple



1. JS Basics
2. The DOM
3. DOM Events

JavaScript Basics

POP QUIZ

+1 Point for Answer

+1 Point for Code Example

Question 1/8

What is a **variable**?

Question 2/8

What is an **array**?

Question 3/8

What is a **function**?

Question 4/8

What is an **object**?

Question 5/8

What is a **loop**?

Question 6/8

What is a **conditional**?

Question 7/8

How do you **include JavaScript in HTML** ?

Question 8/8

How do you **include** **JavaScript** in a WP Theme?

How Did You Do???

POP QUIZ

+1 Point for Answer

+1 Point for Code Example

Variable

A container for storing values (in memory)

```
var username = 'zgordon';
let username = 'zgordon';
const siteURL = 'https://site.com';
```

Array

A collection of values

```
let postIds = [ 1, 2, 3, 5 ];
let usernames = [
      'zgordon',
      'admin'
    ];
```

Array

A collection of values (zero indexed)

```
let postIds = [ 1, 2, 3, 5 ];

postIds[ 0 ] // Equals 1
postIds[ postIds.length - 1 ] // Last item
```

Functions

Let us write, call and reuse blocks of code

```
function getPosts() {
  let posts = apiMagic(); // Will learn
  return posts;
}
getPosts();
```

Function Parameters

Can pass data into functions

```
function getUser( id = 0 ) {
    let user = apiMagic( id );
    return user;
}
getUser( 1 );
```

Objects

Container with properties (values) and methods (functions).

```
let post = {
  'title more': 'Hello World!',
  render: function() {
    console.log( this.title );
  }
}
```

```
post.title;
post.render();
```

Loops

Let us perform an action on a collection of items.

```
let postIds = [ 1, 7, 14, 34, 88, 117 ];

for ( let i = 0, max = postIds.length; i < max; i++ ) {
  console.log( 'Post #' + postIds[ i ] );
}
```

For Of Loop

Lets us loop through an array

```
var postIds = [ 1, 7, 14 ];

for( let id of postIds ) {
  console.log( id );
}
```

Conditional Statements

Tests to determine what code to run

```
let loggedIn = true;
if ( true === loggedIn ) {
  console.log( 'Show dashboard' );
} else {
  console.log( 'Please login' );
}
```

Include JS in HTML

```
<html>
  <head>
  </head>
  <body>
    <script src="index.js"></script>
  </body>
</html>
```

Include JS in WP Theme

```
function theme_scripts() {
  wp_enqueue_script(
              'main-js',
              get_template_directory_uri() . '/js/main.js',
              [ 'jquery' ],
              time(),
              true );
}
add_action('wp_enqueue_scripts', 'theme_scripts', 999);
```

JavaScript & APIs Made Simple

1. ~~JS Basics~~
2. The DOM
3. DOM Events

The DOM

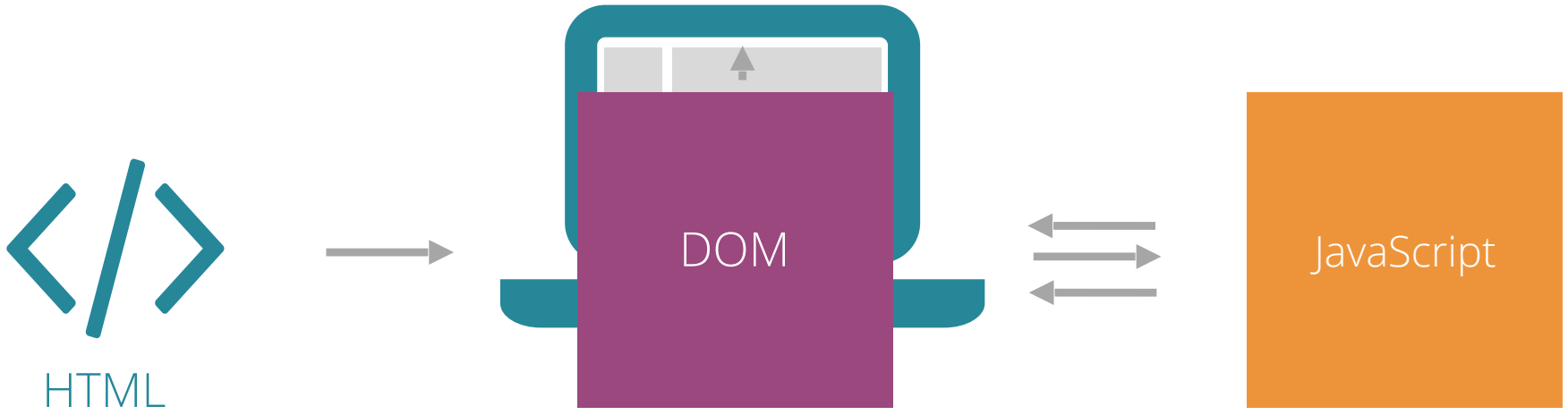
Document Object Model

The DOM is an API

For HTML (XML & SVG)

The DOM is

NOT Source Code


HTML
DOM
JavaScript

Build documents, **navigate** their structure, and **add, modify, or delete** elements and content.

The DOM API

1. Build
2. Navigate
3. Add, Modify, Delete

The DOM API

1. Build Nodes
2. Navigate Selection, Traversal
3. Add, Modify, Delete Append, Set, Remove

Everything is a Node

In the DOM

Common Node Types

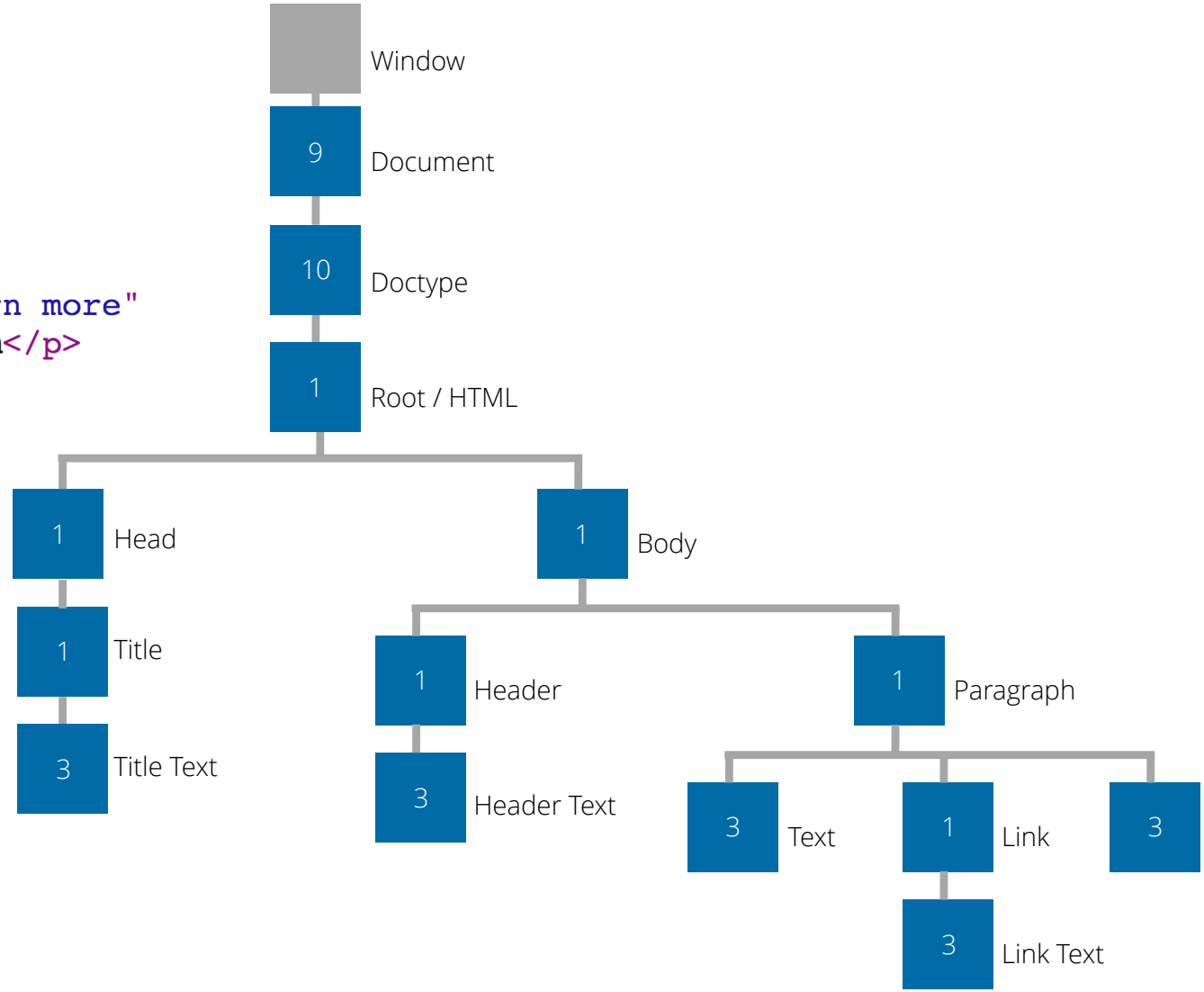
- Document [9]
- DocumentType [10]
- Element [1]
- Text [3]
- Comments [8]
- DocumentFragments [11]

Common Node Types

- Document [9] index.html
- DocumentType [10]
- **Element [1]**
- **Text [3]**
- Comments [8]
- *DocumentFragments [11]*

```
<!DOCTYPE html>
<html>
<head>
  <title>The DOM</title>
</head>
<body>
  <h1>Title</h1>
  <p>Lorem <a title="Learn more"
href="#">to the</a> ipsum</p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
  <title>The DOM</title>
</head>
<body>
  <h1>Title</h1>
  <p>Lorem <a title="Learn more"
href="#">to the</a> ipsum</p>
</body>
</html>
```



Practice 1.1

Looking up Node Types

Document Object Methods

Getting HTML Elements

```
document.getElementById( 'main' )
document.getElementsByTagName( 'a' )
document.getElementsByClassName( 'post' )
document.querySelector( '.entry-title a' )
document.querySelectorAll( '.entry-title a' )
```

Getting Node Values

```
const el = document.getElementById( 'main' );
```

```
el.innerText
el.innerHTML
input.value
el.id
el.href
el.dataset.custom
```

Setting Node Values

```
const el = document.getElementById( 'main' );

el.innerText = 'New';

el.innerHTML = '<p>New</p>';

input.value = 'zgordon';
```

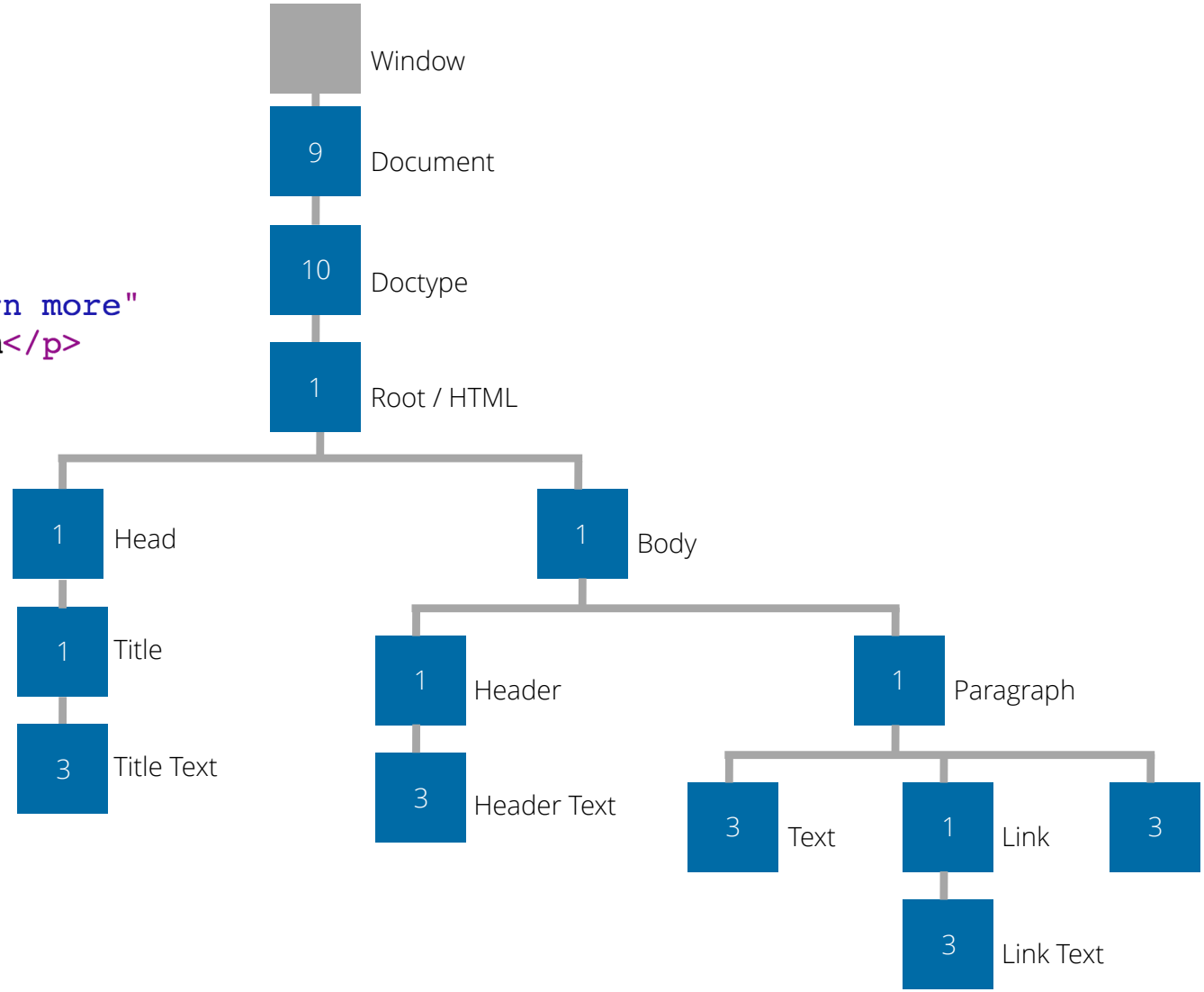
Styling Nodes

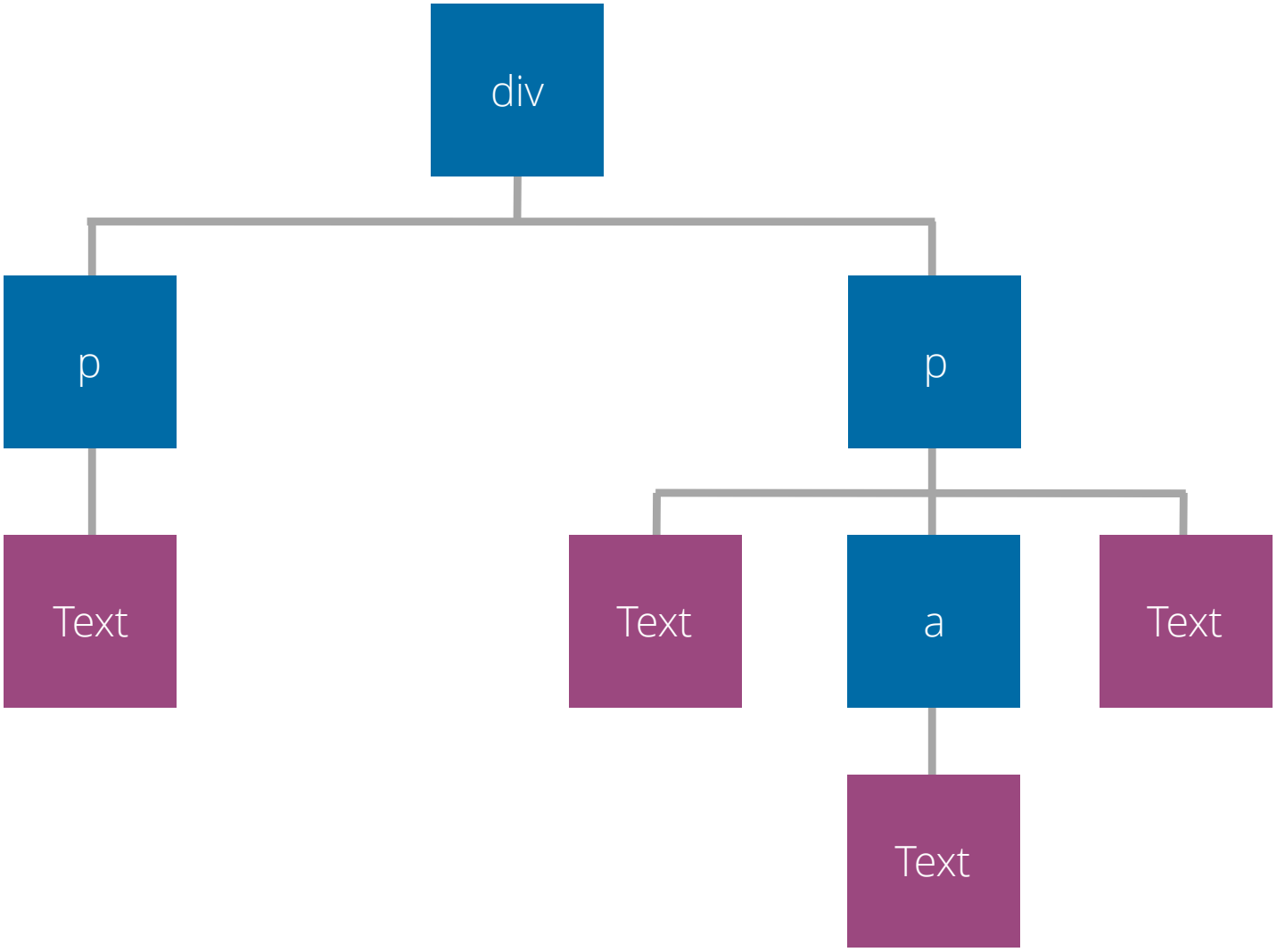
```
const el = document.getElementById( 'main' );

window.getComputedStyle( el )
el.style.backgroundColor = '#f7b733'

el.classList.add( 'new' )
el.classList.remove( 'new' )
el.classList.toggle( 'new' )
```

Practice 1.2
Getting & Setting Values

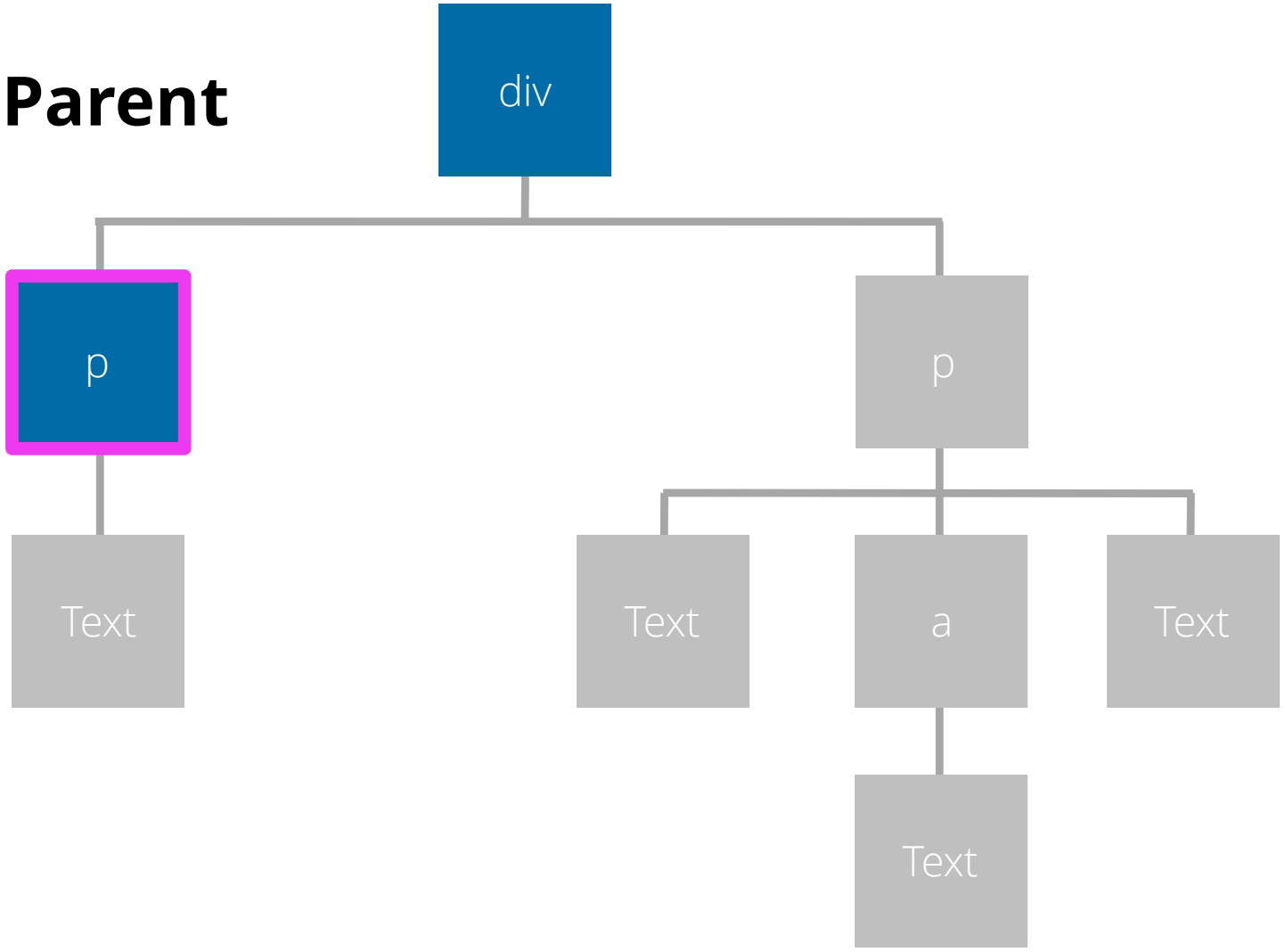
DOM Traversal

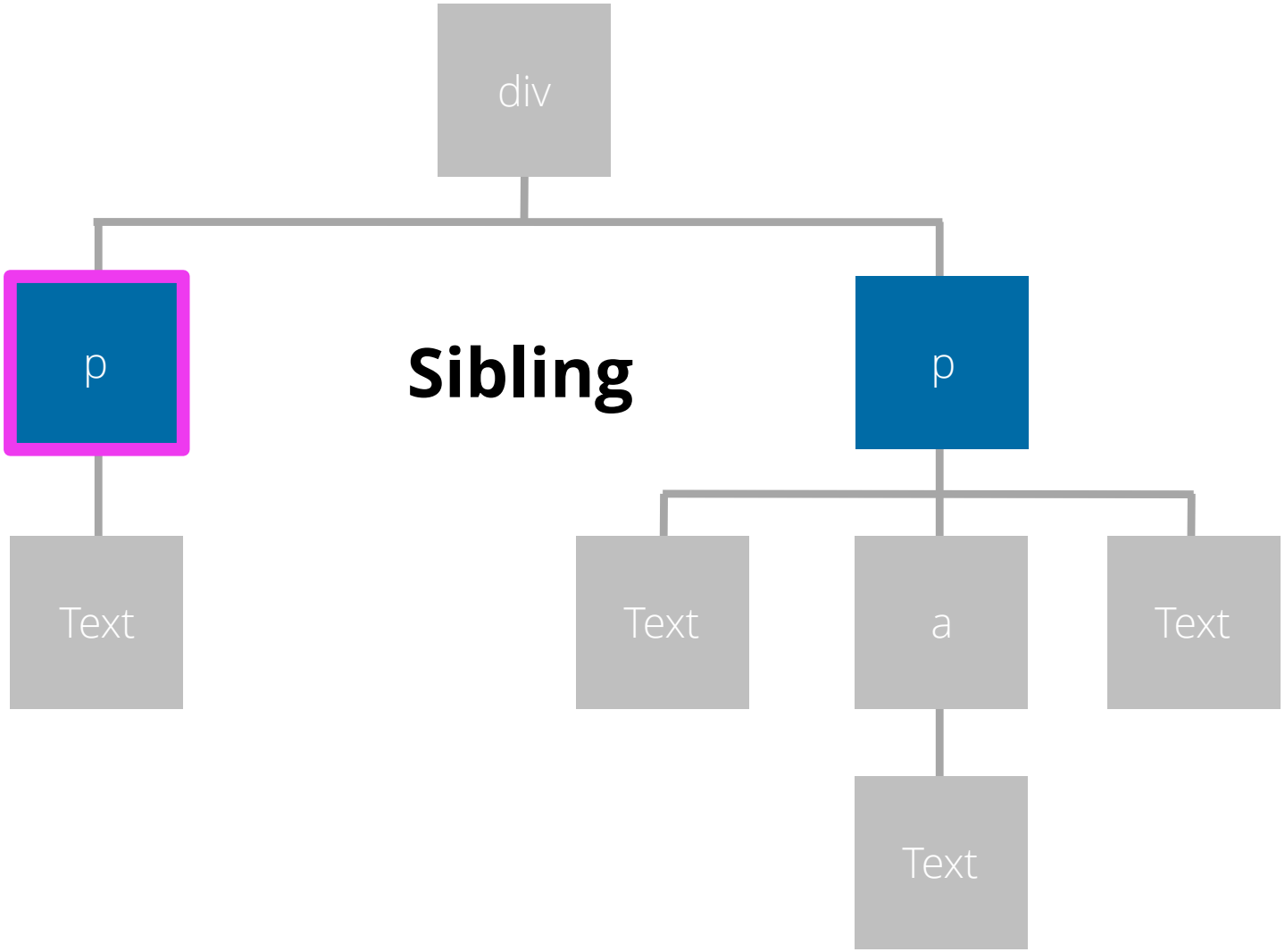
```
<!DOCTYPE html>
<html>
<head>
  <title>The DOM</title>
</head>
<body>
  <h1>Title</h1>
  <p>Lorem <a title="Learn more"
href="#">to the</a> ipsum</p>
</body>
</html>
```



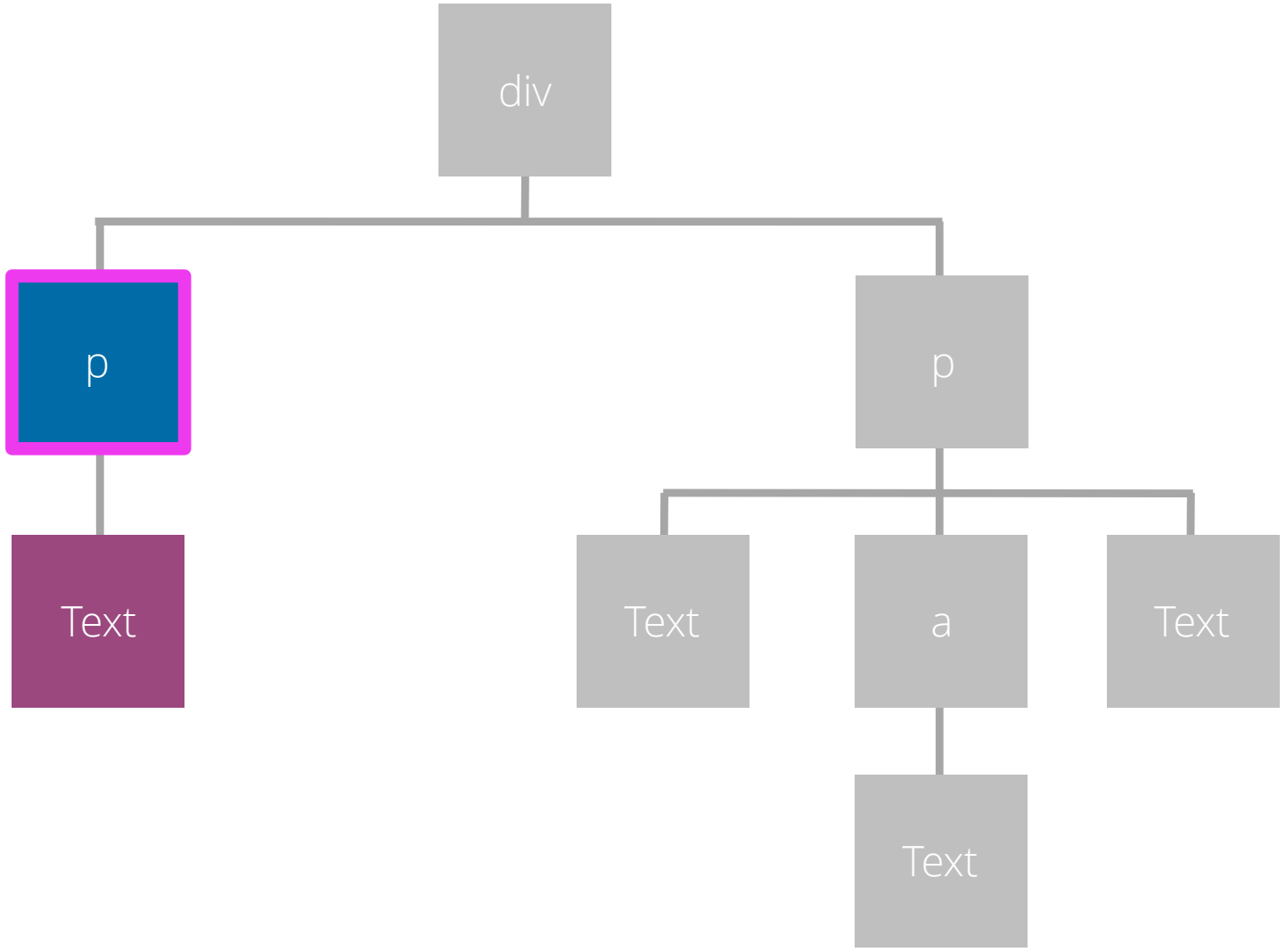

div
p
p
Text
Text
a
Text
Text

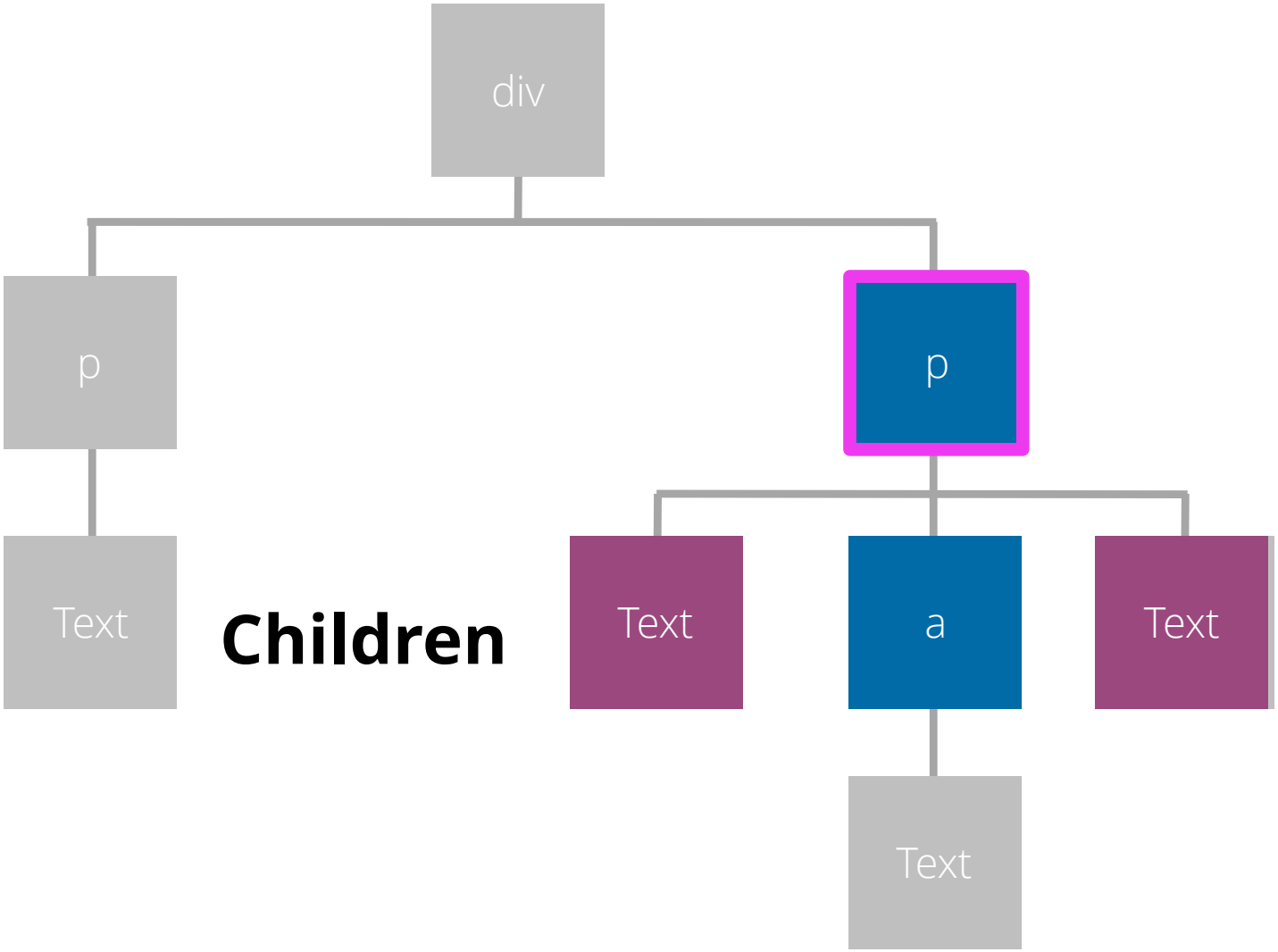
Parent




div
p
Sibling
p
Text
Text
a
Text
Text

Child




div
p
p
Text
Children
Text
a
Text
Text

DOM Traversal

Any Node

```
el.parentNode
el.childNodes
el.firstChild
el.lastChild
el.nextSibling
el.previousSibling
```

Element Nodes

```
el.parentElement
el.children
el.firstElementChild
el.lastElementChild
el.nextElementSibling
el.previousElementSibling
```

Practice 1.3

DOM Traversal

Creating & Appending Nodes

Creating Nodes

```
document.createElement( 'p' )
document.createTextNode( 'Text' )
document.createDocumentFragment()
```

Appending Nodes

```
parent.appendChild( newEl )
parent.insertBefore( newEl, el )
parent.innerHTML = '<p>Text</p>'
```

Appending Nodes

```
let pEl = document.querySelector( 'p' ),
    aEl = document.createElement( 'a' ),
    aText = document.createTextNode( 'Link' );

aEl.appendChild( aText );
aEl.href = 'https://javascriptforwp.com';
pEl.appendChild( aEl );
```

"Fake" Appending Nodes

```
let pEl = document.querySelector( 'p' ),
    markup = '';

markup += '<a href="https://site.com">' );
markup += 'Link Text';
markup += '</a>';

pEl.innerHTML = markup;
```

Practice 1.4

Creating & Appending Nodes

Cloning & Removing Nodes

Cloning Nodes

```
el.cloneNode( true/false )
```

Removing Nodes

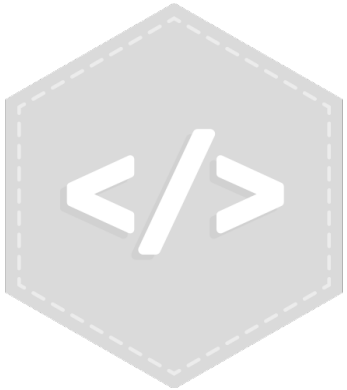
```
el.remove()
el.removeChild()
```

Cloning and Removing

```
let ul = document.querySelector( 'ul' ),
    li = ul.firstElementChild.cloneNode( true );

ul.appendChild( li );
ul.firstElementChild.remove();
```

Practice 1.5
Cloning & Removing

JavaScript & APIs Made Simple



1. ~~JS Basics~~ ~~2. The DOM~~ 3. DOM Events

DOM Events w JavaScript

Types of Events

- Mouse events
- Keyboard events
- Form events
- Media events
- Drag and Drop events
- Window events
- Many more....

https://javascriptforwp.com/intro-to-events/

Hooking into Events w JavaScript

1. **Inline** in HTML `<a onclick="alert('hi')">Alert</a>`
2. **Global** 1 Off `a.onclick = sayHi`
3. **Listeners** Best `a.addEventListener( 'click', sayHi, false )`

Event Listeners

```
const linkEl = document.querySelector( 'a' );

function displayLinkInfo( event ) {

  event.preventDefault();
  console.log( event.target.innerHTML );

}

linkEl.addEventListener( 'click', displayLinkInfo, false );
```

Removing Event Listeners

```
linkEl.addEventListener( 'click', func, false );
linkEl.removeEventListener( 'click', func, false );


linkEl.addEventListener( 'click', function( event ) {
    console.log( event );
}, false );
```

Practice 2.1

Event Listener Practice

The Event Object

Event Listeners

```
link.addEventListener(  'click',  sayHi, false );

function sayHi( event ) { // event is the event object
  console.log( event );
  console.log( event.type );
  console.log( event.target );
}
```

Practice 2.2

Checking Out the Event Object

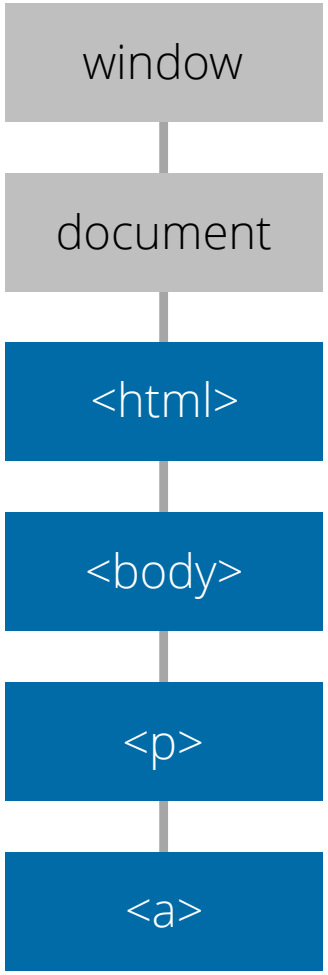
Event Propagation

Event Propagation

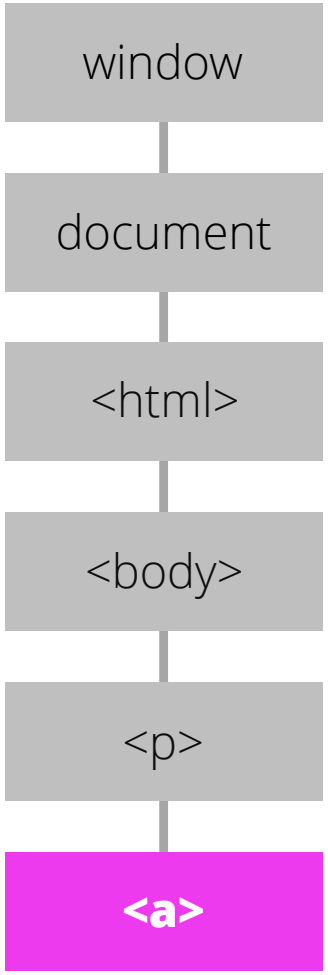
How notifications of events spreads through the DOM

javascriptforwp.com/event-propagation

Event Propagation

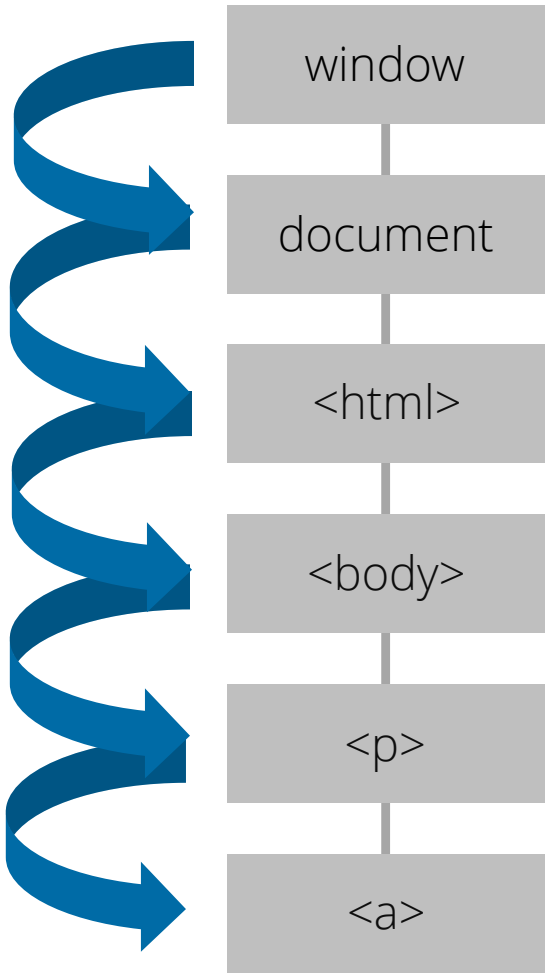


Event Propagation



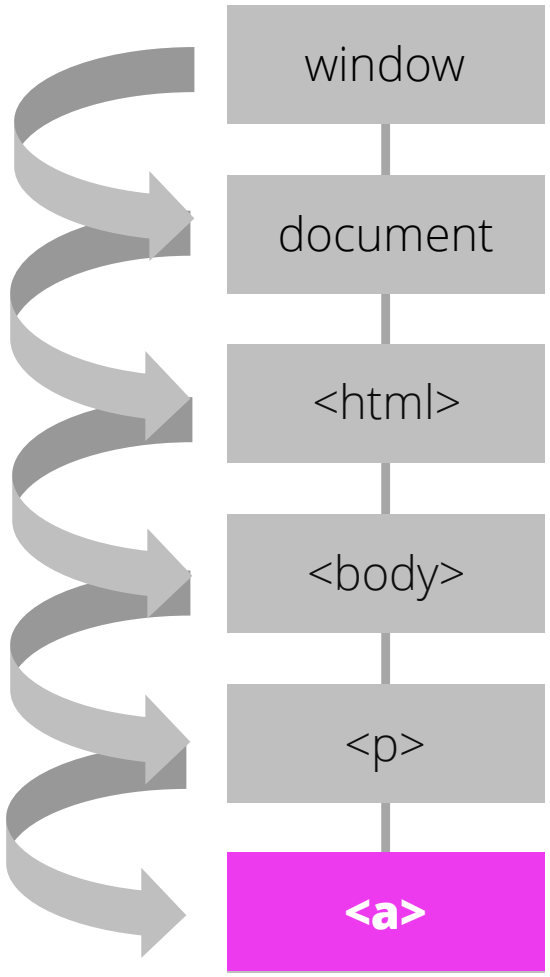
Event Propagation

Phase 1 - Capturing



Event Propagation

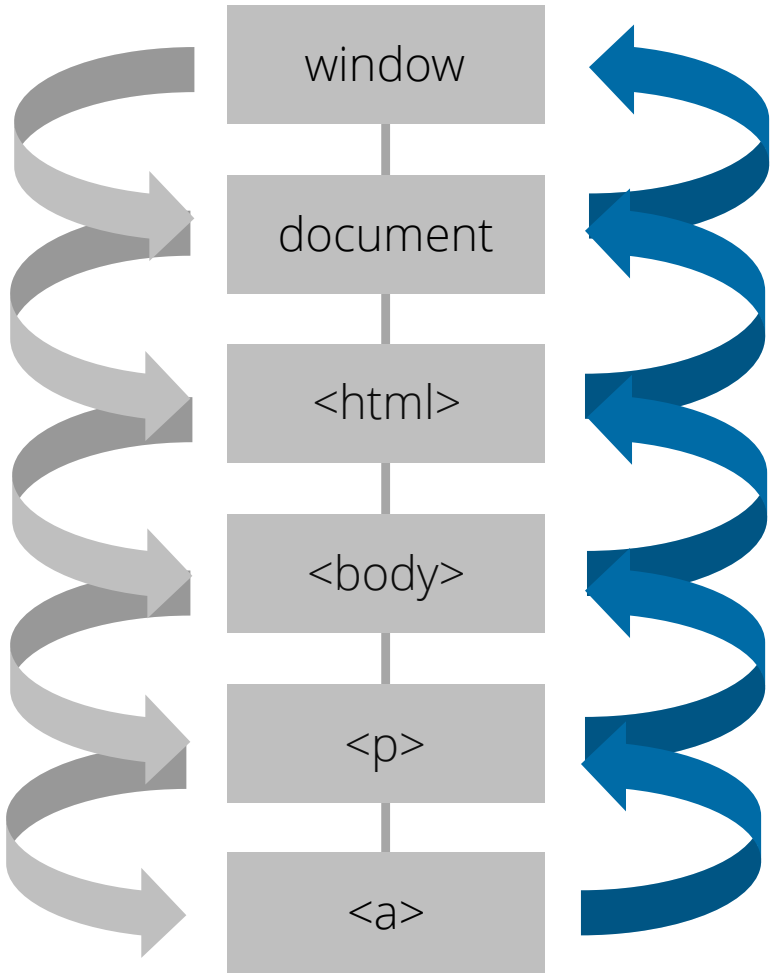
Phase 1 - Capturing



Phase 2 - Target

Event Propagation

Phase 1 - Capturing

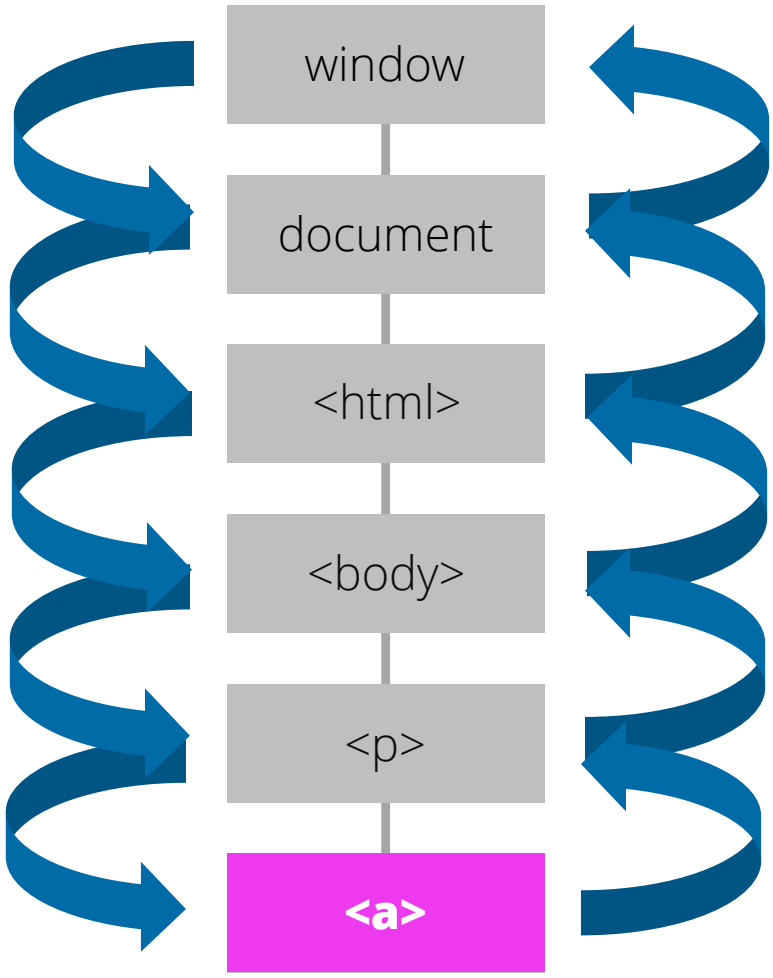


Phase 3 - Bubbling

Phase 2 - Target

Event Propagation

Phase 1 - Capturing



Phase 3 - Bubbling

Phase 2 - Target

Event Propagation

Bubbling

```
el.addEventListener( 'click', sayHi, false );
```

Capturing

```
el.addEventListener( 'click', sayHi, true );
```

Stopping Propagation

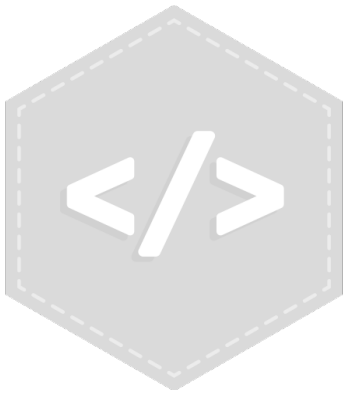
```
event.stopPropagation();
```

Practice 2.3

Not Going to Practice ;)

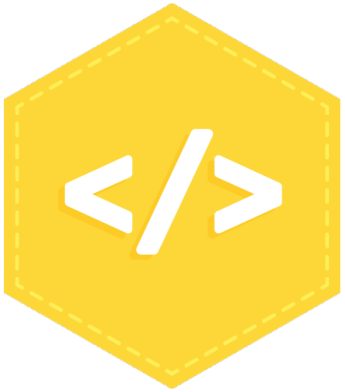
javascriptforwp.com/event-propagation

JavaScript & APIs Made Simple



1. ~~JS Basics~~ ~~2. The DOM~~ ~~3. DOM Events~~

JavaScript & APIs Made Simple



1. JS Basics
2. The DOM
3. DOM Events

It's not that JavaScript is a hard or weird language. However, almost everything you do in JavaScript involves an API. That makes things more complicated.

Final Practice / HW

Pulling it All Together

javascriptforwp.com/category/free-videos

Learn JavaScript Deeply with Zac Gordon
@zgordon