JavaScript for WordPress

Zac Gordon @zgordon

https://javascriptforwp.com

Go Here to Get Setup

javascriptforwp.com/nyc

Are We Ready?!

1. **Slides and Example Files**
2. **Code Editor** (I'm using Atom)
3. **Local WordPress Install** (I'm Using Local)

Workshop Overview

1. **JavaScript** [JS 101, DOM, Events]
2. **The WordPress REST API** [Accessing, Arguments]
3. **JavaScript & WP API in Themes** [Enqueuing, jQuery, API]
4. *Homework!*

JS 101
Pop Quiz!

JS 101 Pop Quiz!!!

1. What is a variable?
2. What is an array?
3. What is a function?
4. What is an Object?
5. What is a Loop?
6. What is a Conditional
7. How do you include JavaScript in HTML?
8. How do you include JavaScript in a WordPress Theme?

JS 101

Variable

A container for storing values (in memory)

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

keyword	var	let	const
global scope	YES	NO	NO
function scope	YES	YES	YES
block scope	NO	YES	YES
can be reassigned	YES	YES	NO

Variable

A container for storing values (in memory)

```
var username = 'zgordon',
    twitter = '@zgordon',
    website = 'zacgordon.com';
```

Array

A collection of values

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

Array

A collection of values (zero indexed)

```
var 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() { // Function Declaration
  var posts = apiMagic(); // Will learn..
  return posts;
}
getPosts(); // Function Call
getPosts // Function Reference
```

Function Parameters

Can pass data into functions

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

Objects

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

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

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

Loops

Let us perform an action on a collection of items.

```
var 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 );
}
```

For Each Method

Lets us loop through an array

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

postIds.forEach( id => {
  console.log( id )
} );
```

Mapping

Mapping is a functional alternative to looping

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


postIds.map( id => {
  console.log( id );
});
```

Conditional Statements

Tests to determine what code to run

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

Include JS in HTML

```
<html>
  <head>
  </head>
  <body>
        <script>
            console.log( 'JS Running!' );
        </script>
  </body>
</html>
```

Include JS in HTML

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

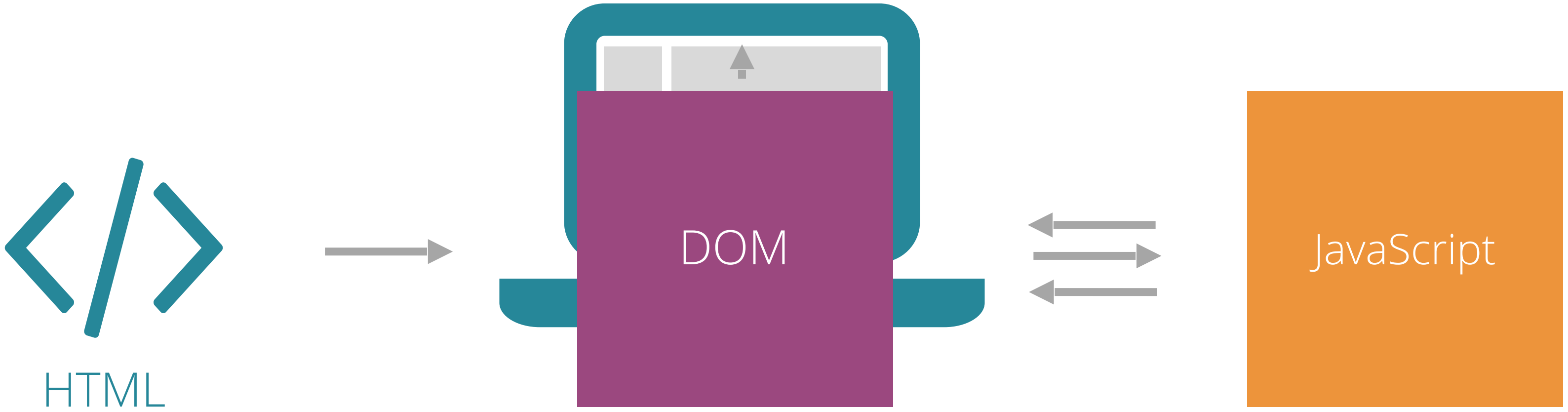
Include JS in WP Theme

```
function my_theme_scripts() {
  wp_enqueue_script(
      'main-js',
      get_template_directory_uri() . '/js/main.js',
      [ 'jquery', 'wp-api' ],
      time(),
      true );
}
add_action( 'wp_enqueue_scripts', ‘my_theme_scripts' );
```

The DOM & Nodes

The DOM is an API

For HTML (XML & SVG)


HTML
DOM
JavaScript

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

Everything in the DOM

Is a Node

Common Node Types

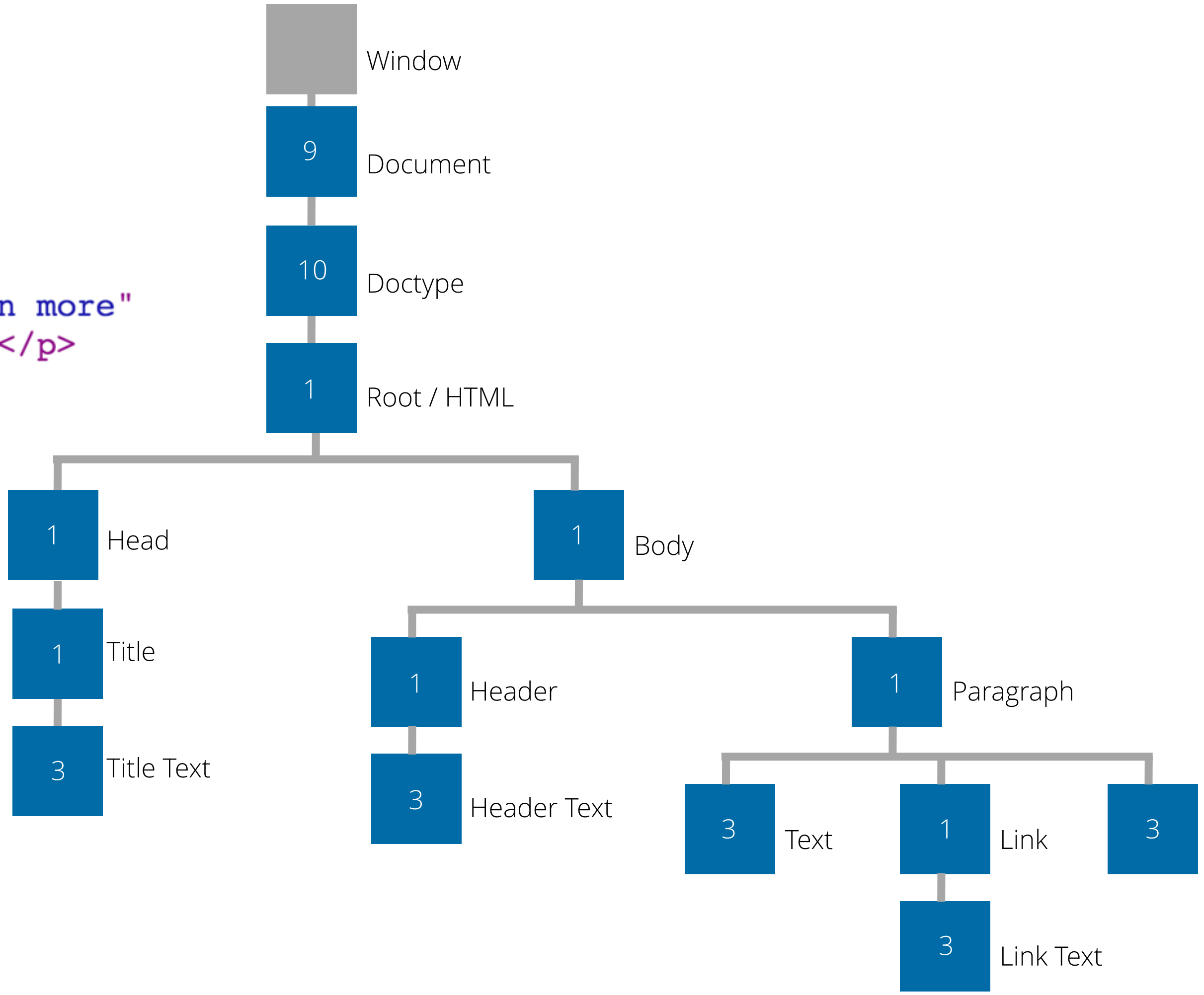
- Document [9]
- DocumentType [10]
- Element [1]
- Text [3]
- Comments [8]
- DocumentFragments [11]

Common Node Types

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

```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>
```

```
<!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>
```



Getting DOM Nodes w JS

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

Getting Node Values

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

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

Setting Node Values

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

el.innerText = 'New';
el.innerHTML = '<p>New</p>';
input.value = 'zgordon';
```

PRACTICE 1.1

Getting & Setting Nodes w JS

Creating & Appending Nodes

Creating Nodes

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

Appending Nodes

```
parent.appendChild( newEl );
parent.insertBefore( newEl, el );
```

Appending Nodes Example

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

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

“Hacker Style” Appending Nodes

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

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

pEl.innerHTML = markup;
```

Practice 1.2 + 1.3

Creating & Appending Nodes

Events w JavaScript

Types of Events

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

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

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

function displayLinkInfo( event ) {

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

}

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

The Event Object

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

function sayHi( event ) {
  console.log( event );
  console.log( event.type );
  console.log( event.target );
}
```

Removing Event Listeners

```
var el = document.querySelector( 'a' );

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

Removing Event Listeners

```
var el = document.querySelector( 'a' );

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

el.remove(); // Listener still exists
el = null;  // Listener does NOT exist
```

Practice 1.4

Event Listeners Practice

Workshop Overview

1. ~~JavaScript [JS 101, DOM, Events]~~
2. **The WordPress REST API [Accessing, Arguments]**
3. JavaScript & WP API in Themes [Enqueuing, jQuery, API]
4. *Homework!*

The WordPress REST API

The WP REST API

Allows us to get, save, edit, and remove content from WordPress using JavaScript.

The WordPress REST API

```
/wp-json/wp/v2/posts         /wp-json/wp/v2/taxonomies
/wp-json/wp/v2/revisions     /wp-json/wp/v2/media
/wp-json/wp/v2/categories    /wp-json/wp/v2/users
/wp-json/wp/v2/tags          /wp-json/wp/v2/types
/wp-json/wp/v2/pages         /wp-json/wp/v2/statuses
/wp-json/wp/v2/comments      /wp-json/wp/v2/settings

        /wp-json/custom/v1/something
        /wp-json/another/v3/else
```

The WordPress REST API

To access all of the posts for your site:

https://yoursite.com/wp-json/wp/v2/posts

To access a **post with an ID** of “1” for your site:

https://yoursite.com/wp-json/wp/v2/posts/1

The WordPress REST API

To access all of the pages for your site:

https://yoursite.com/wp-json/wp/v2/pages

To access a **page with an ID** of “5” for your site:

https://yoursite.com/wp-json/wp/v2/pages/5

You can pass **arguments** to API URLs to **customize what you get** back.

The WordPress REST API

```
https://yoursite.com/wp-json/wp/v2/posts
        ? per_page=10
        &_embed
        &search=hello%20world
        &orderby=title
        &order=asc
```

https://developer.wordpress.org/rest-api/reference/posts/#arguments

Practice 2.1

WP API in the Browser

WordPress API
w AJAX + JS

ajax

Allows for us to make calls to the WP REST API and get back content in a format JavaScript can understand. (Can also save, edit and delete)

Ways to Make JS AJAX Calls

- XMLHttpRequests()
- fetch()
- jQuery
- Axios
- API Clients (Backbone, Node)

AJAX Behind the Scenes

1. Request a URL
2. Get JSON in return
3. Parse JSON into Native JavaScript
4. Use Native JavaScript

Pseudo AJAX

```
var postsJSON = fetch('https://wp.dev/wp-json/wp/v2/posts'),
     posts = JSON.parse( postsJSON );

for( let post of posts ) {
   renderPost( post );
}
```

AJAX with Axios

```
axios({
    method: 'get',
    url: 'https://example.dev/wp-json/wp/v2/posts',
    params: {
        per_page: 3
    }
})
    .then( function( response ) {
        for( let post of response.data ) {
            renderPost( post );
        }
    });
```

Practice 2.2 + 2.3

WP API with Axios Ajax

Workshop Overview

1. ~~JavaScript [JS 101, DOM, Events]~~
2. ~~The WordPress REST API [Accessing, Arguments]~~
3. **JavaScript & WP API in Themes [Enqueuing, jQuery, API]**
4. *Homework!*

JS + API
Inside of WP

Enqueuing JavaScript

The process of loading JavaScript into a theme or the admin area via PHP functions.

wp_enqueue_script()

wp_localize_script()

wp_enqueue_script()

```
function my_theme_scripts() {
  wp_enqueue_script(
       'main-js', // Unique name
       get_template_directory_uri() . '/js/main.js', // Path
       [ 'jquery', 'wp-api' ], // Dependencies
       time(), // Version
       true ); // Include in footer (v header)
}
add_action('wp_enqueue_scripts', 'my_theme_scripts', 999);
```

wp_localize_script()

```
function my_theme_scripts() {
  wp_enqueue_script( 'main-js', ... );
  wp_localize_script(
    ‘main-js',  // File to localize
    ‘myGlobalVars', // Name of localize object
    [  // Array of properties
      'siteUrl' => esc_url( home_url() ),
      'siteName'=> get_bloginfo( 'name' )
    ]
  );
}
add_action('wp_enqueue_scripts', 'my_theme_scripts', 999);
```

wp_localize_script()

functions.php

```
function my_theme_scripts() {
  wp_enqueue_script( 'main-js', ... );
  wp_localize_script( 'main-js', 'myGlobalVars', ... );
}
add_action('wp_enqueue_scripts', 'my_theme_scripts', 999);
```

main.js

```
console.log( myGlobalVars.siteUrl );
```

Enqueuing JS in Admin Area

```
function my_theme_scripts() {

  wp_enqueue_script( 'main-js', ... );
  wp_localize_script( 'main-js', 'myGlobalVars', ... );

}

add_action( 'wp_enqueue_scripts', 'my_theme_scripts', 999);
add_action( 'admin_enqueue_scripts', 'my_theme_scripts', 999 );
```

Enqueuing JS in Admin Area

```
function my_admin_scripts( $hook ) {

  global $main_menu_url;

  if( $hook != 'toplevel_page_' . $main_menu_url ) return;

  wp_enqueue_script( 'my-admin-js', plugins_url( '/assets/js/
admin.js', __FILE__ ), [ 'wp-api' ], time(), true );

}
add_action( 'admin_enqueue_scripts', 'my_admin_scripts', 999);
```

Practice 3.1

Enqueuing JavaScript in a Theme

jQuery + WordPress

WordPress comes with

Several **JS Libraries**

Available to Us

jQuery Dependency

```
function my_theme_scripts() {
  wp_enqueue_script(
       'main-js', // Unique name
       get_template_directory_uri() . '/js/main.js', // Path
       [ 'jquery' ], // Dependencies
       time(), // Version
       true ); // Include in footer (v header)
}
add_action('wp_enqueue_scripts', 'my_theme_scripts', 999);
```

The **$** in jQuery

is **NOT available**

by **default**

```
jQuery( ‘.this-works' ).show();
```

Immediately Evoked Function Expression (IEFE)

```
(function( $ ) {

  $('.this-works').show();

})( jQuery );
```

Practice 3.2

Using jQuery for AJAX

The Backbone WP API Client

Backbone API Client

Allows us to easily use the WP REST API in a theme or plugin without writing AJAX or authentication.

*No knowledge of Backbone necessary**

Backbone API Client Dependency

```
function my_theme_scripts() {
  wp_enqueue_script(
       'main-js', // Unique name
       get_template_directory_uri() . '/js/main.js', // Path
       [ ‘wp-api’ ], // Dependencies
       time(), // Version
       true ); // Include in footer (v header)
}
add_action('wp_enqueue_scripts', 'my_theme_scripts', 999);
```

Backbone Client - Get Posts

```
var posts = new wp.api.collections.Posts();
    posts.fetch({
        data: {
            per_page: 3
        }
    })
       .done( () => {
           posts.each( post => renderPost( post.attributes );
        );
```

Backbone Client - Get Post By ID

```
var post = new wp.api.models.Post( { id } );

post.fetch()
    .done( () => renderPost( post.attributes ) );
```

Practice 3.3

Using the Backbone API Client

Workshop Overview

1. ~~JavaScript [JS 101, DOM, Events]~~
2. ~~The WordPress REST API [Accessing, Arguments]~~
3. ~~JavaScript & WP API in Themes [Enqueuing, jQuery, API]~~
4. ***Homework!***

Thanks!

javascriptforwp.com/wcnyc

Learn JavaScript Deeply with Zac Gordon

@zgordon