آموزش ایجکس در وردپرس

Ajax in WordPress

نویسنده: مهرشاد درزی

کارشناس ، مشاور و توسعه دهنده وردپرس

وب سایت: RealWp.Net

ایجیکس (**Ajax**) یکی از فناوری های ارتباط هماهنگ هست که در بسیاری از قالب ها و افزونه های وردپرس که در مارکت وجود دارد استفاده می شود.یک **توسعه دهنده وردپرس** در درجه ی اول میبایست سرعت انجام عملیات کاربر را افزایش دهد.یکی از روش هایی که به تعامل سریع تر و زیباتر پروژه های وردپرس کمک فراوانی می کند، استفاده از فناوری ایجکس می باشد.

متاسفانه تا به امروز قالب ها و افزونه های زیادی را مشاهده کردم که اصلا در برنامه نویسی های آن به مستندات وردپرس توجهی نداشته اند. و روش های اشتباهی را برای ایجاد ساختار دستورات Ajax در وردپرس به کار گرفته اند.نتیجه نهایی آن کاهش سرعت و افزایش حجم ریکوئست ها و درخواست به سمت سرور هاست وردپرس شد و در نهایت نتیجه گیری در ذهن مخاطب که وردپرس بدرد نمیخورد.

چرا از ایجکس در وردپرس استفاده می کنیم؟

به صورت کلی ۳ دلیل اصلی برای استفاده از **ایجکس در وردپرس** وجود دارد:

استفاده از افزونه های کش وردپرس

زمانی که ما در وردپرس از افزونه های کش (WordPress Cache) استفاده می کنیم. دیگر صفحات به صورت داینامیک بارگزاری نمی شوند. یعنی زمانی که شما اولین بار صفحه ای از یک سایت وردپرس را در مرورگر باز می کنید. آن صفحه در وردپرس کش می شود و برای دیگر بازدید کنندگان جهت افزایش سرعت از نسخه ی کش شده بارگزاری می شود.

حال فرض کنید شما در صفحه ی خود بخشی دارید که می بایست داینامیک باشد. مثلا می خواهید محصولات فروشگاه به شکل تصادفی نمایش داده شود ، یا تاریخ امروز به شمسی در بالای سایت پدیدار شود.در این حالت یکی از بهینه ترین روش ها اینگونه هست که برای این بخش ها ما از ایجکس (Ajax) در وردپرس استفاده کنیم تا شامل کش سرور نباشد.

افزونه های کش وردپرس مانند LiteSpeed یا WP Rocket یا W3 Total Cache و .... در حالت عادی درخواست های آجاکس و REST API وردپرس را در فایل ذخیره یا کش نمی کنند

سرعت بارگزاری سریع تر وب سایت وردپرس

در ایجکس وردپرس شما این قابلیت را دارید که فقط بخشی از صفحه را مجدد با محتوا و داده های جدید بارگزاری کنید، این کار باعث می شود که صفحه در مرورگر کاربر رفرش نشود. این کار ن تنها لذت کار با سایت شما را برای بازدید کننده افزایش می دهد. بلکه سرعت عملکرد پروسس وردپرس نیز بالاتر می رود. چون وردپرس در این حالت موظف هست آن بخش به خصوص را مجدد بارگزاری کند ن کل صفحه را.

یکی از مثال های بارز ایجکس در وردپرس برای افزونه ووکامرس ، قسمت سبد خرید یا Cart هست که در اکثر قالب بعد از فشردن دکمه های افزودن به سبد خرید، کاربر به صفحه ی دیگر هدایت نمی شود بلکه سبد خرید کاربر در ووکامرس بروز رسانی می شود.

البته نا گفته نماند افزونه ووکامرس اشتباهاتی را در بخش ایجیکس وردپرس انجام داده که اگر مقاله را تا پایان بخوانید متوجه این موضوع خواهید شد.

جداسازی بخش فرانت و بک اند پروژه وردپرس

یکی از اصولی ترین روش های برنامه نویسی و توسعه استارتاپ چه در بستر وردپرس یا هر فریم ورک دیگری مثل لاراول و .. ، جداسازی بخش فرانت اند (Front-End) و بک اند (Back-End) می باشد. ما بوسیله ی ایجاد درخواست های Ajax در وردپرس این قابلیت را به آن اضافه می کنیم که از داده های وردپرس خود ن تنها در قالب سایت ، بلکه در هر پلتفرم دیگری نیز استفاده کنیم.

من در این مقاله یک مثال از این روش برای توسعه استاندارد و طراحی سایت های وردپرس زدم که حتما مطالعه کنید:

حتما بخوانید : برنامه نویسی قالب وردپرس بر اساس متد یکسازی سازی REST API وردپرس

انواع روش های ایجکس در وردپرس چیست؟

3روش اصلی برای درخواست های ایجکس (Ajax) در وردپرس وجود دارد که هر کدام دارای معایب و مزایای خودش هست:

روش اول ایجاد درخواست ایجکس در وردپرس بوسیله ی فایل **admin-ajax.php** می باشد، این روش عمومی ترین و در عین حال بدترین روش ارسال درخواست های ایجکس در وردپرس می باشد که اکثر قالب ها و افزونه ها طبق آموزش های سطح پایین در وب از آن استفاده می کنند.

روش دوم استفاده از وب سرویس (**REST API**) وردپرس می باشد. با توجه به پیشرفت روز افزون بحث توسعه ی اپلیکیشن ها و WordPress Headless ، بسیاری از پلاگین ها مانند Contact Form 7یا Yoast Seo درخواست های خود را به سبک REST API بهینه کرده اند.

روش سوم استفاده از متد **Rewrite API** هست که میتوان آن را بهینه ترین حالت برای ارسال درخواست ایجکس سمت کاربر دانست.در این روش با ایجاد یک آدرس اختصاصی در رده ی اول ریکوئست های وردپرس آن را ملزم به فراخوانی داده ها می کنیم.

مقایسه عملکرد سرعت روش های Ajax در وردپرس

اگر به فایل wp-settings.php در هسته ی وردپرس که فایل مرجع برای لود تمامی بخش ها و توابع هسته ی وردپرس هست دقت کنیم. خواهیم دید که بخش Rewrite مرتبط با initialization وردپرس می باشد یعنی قسمتی که ابتدا فراخوانی می شود. سپس بعد از ضمیمه شدن کلاس کوئری ها در وردپرس به بخش فایل های REST API خواهیم رسید.

اگر همین امر را مبنا قرار دهیم سرعت دسترسی وردپرس به بخش Rewrite سریع از دو مورد دیگر انجام می شود. این خود میتواند یکی از دلایل اصلی ارجحیت ایجکس وردپرس بر اساس سیستم پیوند های یکتای مرکزی یا Rewrite باشد.در تست هایی که توسط توسعه دهندگان وردپرس در تعداد درخواست زیاد صورت گرفته است استفاده از ایجکس در وردپرس بر اساس روش Rewrite میتواند ۴۰٪ سرعت درخواست های وردپرس را بهبود ببخشد.

پس در یک جمع بندی برای روش درخواست ای جکس در وردپرس خواهیم داشت:

قوانین پیش فرض درخواست های ایجکس در وردپرس

دو قانون را میبایست در زمان برنامه نویسی درخواست Ajax در وردپرس رعایت کنید. زیرا اشتباه در هر کدام می تواند نتیجه ی عکس برای شما در بر داشته باشد.

استفاده از فرمت داده های json در خروجی ایجیکس وردپرس

تمامی داده هایی که قرار است از درخواست ایجکس وردپرس دریافت شود میبایست بر اساس فرمت داده های رشته String و یا json باشد.اشتباهی که بسیاری از افزونه ها و قالب های وردپرس انجام می دهند ارائه خروجی های HTML در بستر ایجکس وردپرس می باشد. این کار صدمه ی شدید به رویکرد درخواست می زند.این اشتباه مهلک حتی در افزونه هایی مانند ووکامرس (WooCommerce) هم گاها به چشم می خورد.

کاربر را معطل انجام پروسس های وردپرس نکن

یادتان که نرفته هدف اولیه ما برای ایجاد درخواست های ایجکس راحتی UX و لذت استفاده از وب سایت یا اپلیکیشن وردپرسی ما می باشد. به هیچ عنوان نبایستی عملیات سنگینی در درخواست های ایجکس وردپرس انجام شود و کاربر چندین ثانیه معطل یک پاسخ از سمت وردپرس باشد.

در هر کجای درخواست ایجکس اگر لحظه ای گمان کردید که احتمال دارد در بازدید های زیاد کاربر معطل بماند میبایست پیام موفقیت آمیز بودن یا نبودن را ابتدا نمایش دهید و کار را به بخش Cronjobیا پشت صحنه وردپرس واگذار کنید.

فرض کنید می خواهید بعد از ثبت نام کاربر یک ایمیل به آن بزنید ، بهترین حالت برنامه نویسی این است که پیام موفقیت آمیز بودن ثبت را کاربر دریافت کند، سپس یک Scheduled به وردپرس اضافه کند تا در سریع ترین زمان ممکن توسط تابع wp_mail ایمیل را ارسال کند. چون شاید در یک مواقع SMTPسرور با خطا روبرو شود برای همین کاربر چند ثانیه معطل می ماند که ایمیل ارسال شود و این کار جایز نیست.

ایجاد درخواست ایجکس توسط متد admin-ajax در وردپرس

تمامی درخواست ها به فایل مرکزی admin-ajax.php ارسال می شوند. هر درخواست دارای یک شناسه به خصوص می باشد که میبایست به صورت پارامتر action به این فایل ارسال شود.فرض کنید میخواهیم یک درخواست ایجکس در وردپرس ایجاد کنیم برای دریافت اطلاعات پست با شناسه ی ۱ در وردپرس و میخواهیم نام آن را get_post_detail بگذاریم.

در این حالت آدرس درخواست ایجکس ما در وردپرس، با فرض اینکه نام دامنه ی ما site.com باشد:

```
http://site.com/wp-admin/admin-ajax.php?action=get_post_detail
```

فایل admin-ajax.php در پوشه ی wp-admin وردپرس وجود دارد.چقدر عجیب است ن؟

نحوه ی ایجاد درخواست admin-ajax در وردپرس

دو هوک وردپرس وجود دارد تا بتوان درخواست های ایجکس به سبک admin-ajax را به هسته ی وردپرس معرفی کرد:

```
add_action( 'wp_ajax_{name}', 'function_name' );

add_action( 'wp_ajax_nopriv_{name}', 'function_name' );
```

در بخش {name} میبایست نام درخواست خود را وارد کنیم.در مثال فعلی ما نام درخواست ایجکس get_post_detail میباشد پس خواهیت داشت:

```
add_action( 'wp_ajax_get_post_detail', 'function_name' );
add_action( 'wp_ajax_nopriv_get_post_detail', 'function_name' );
```

تفاوت حالت no private و private در چه می باشد؟

شما نیازی ندارید که هر دو هوک را برای ایجاد یک درخواست ایجکس در وردپرس بنویسید admin. ajaxهمان طور که از اسمش پیداست، برای ایجاد درخواست های ایجکس در محیط مدیریت وردپرس هست. اما متاسفانه قالب ها در بخش فرانت نیز از آن استفاده میکنند.

از آن جا که مدیریت وردپرس به صورت پیش فرض برای کسانی نمایش داده خواهد شد که در سایت وردپرس لاگین (Login) هستند. پس هوک شماره دو یعنی no private را زمانی استفاده می کنیم که می خواهیم یک درخواست را تمامی کاربران وردپرس چه آنهایی که لاگین هستند و چه نیستند مشاهده کنند.

اما اگر از هوک شماره ی یک به تنهایی استفاده کنیم خروجی درخواست ایجکس برای افرادی که لاگین نیستند با خطا مواجه خواهد شد.توضیحات تکمیل تری در مستندات وردپرس اینجا در زمینه ی Ajax in WordPress وجود دارد.

تابع خروجی داده ها بر اساس json در وردپرس

قبلا بیان کردیم که حالت استاندارد خروجی Ajax در وردپرس بر اساس ساختار و فرمت Json می باشد. در هسته ی وردپرس دو تابع وجود دارد به نام wp_send_json_success و wp_send_json_error که می توان برای خروجی داده ها در Ajax استفاده کرد.

تفاوت این دو تابع زمانی است که شما میخواهید موفقیت یا ناموفق بودن عملیات را به سیستم ارائه دهید.اگر عملیات ایجکس در وردپرس موفق بود مسلما از تابع Success استفاده می کنیم و اگر ناموفق بود از تابع Error.این دو تابع در مقدار **Status Code** و پارامتر Success که به صورت boolean می باشند، تفاوت دارند.

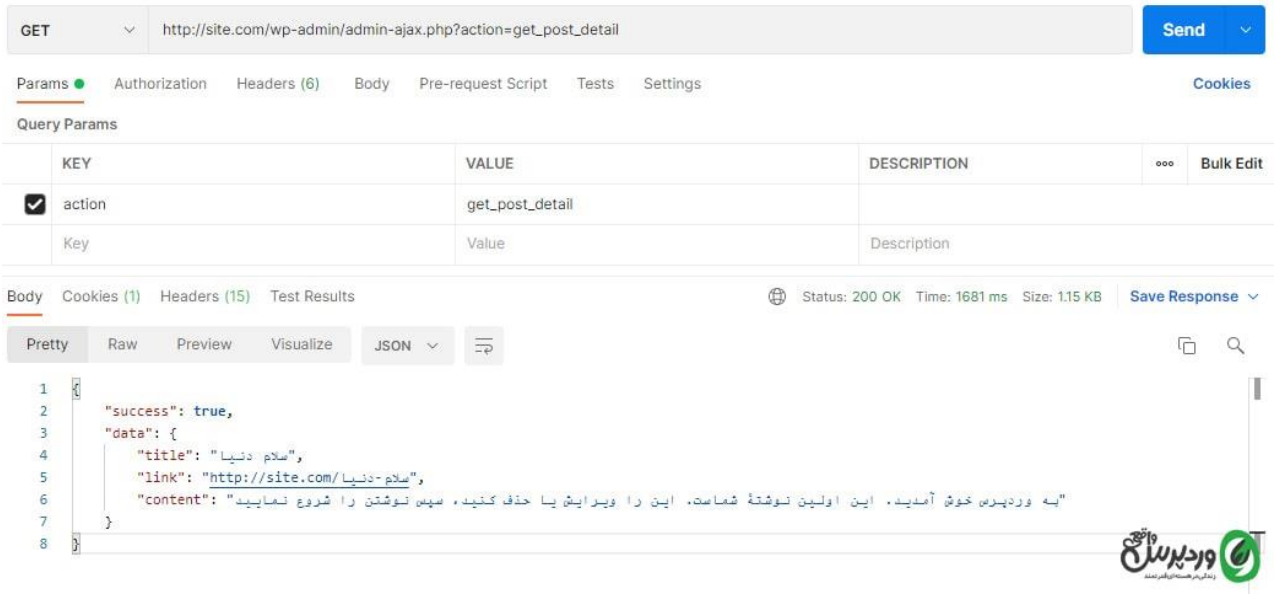
حال بیایید مثال خود را تکمیل کنیم. قرار است اطلاعات پست شماره یک را برای تمامی کاربران نمایش دهیم پس خواهیم داشت:

```
add_action('wp_ajax_nopriv_get_post_detail', 'ajax_get_post_detail');
function ajax_get_post_detail()
{
	$post = get_post(1);
	$array = [
		'title' => $post->post_title,
		'link' => get_the_permalink(1),
		'content' => $post->post_content
	];


	wp_send_json_success($array, 200);

}
```

اگر آدرس بالا را در مرورگر یا نرم افزار POSTMAN تست کنیم. خروجی داده ها به این شکل خواهد شد:



دقت کنید که ساختار تابع wp_send_json_success به دو قسمت success که وضعیت عملیات و data ، مقدار های بازگشتی تقسیم می شود.همچنین میتوانیم Status Code های دلخواه خود را به این تابع اضافه کنیم.

حال که برنامه نویسی سمت وردپرس تکمیل شد میتوانید در قسمت فرانت با هر فریم ورکی که دوست دارید آن را بارگزاری کنید. مثلا اگر با پیش فرض کتابخانه **JQuery**کار می کنید خواهیم داشت:

```
jQuery(document).ready(function ($) {
		$.ajax({
			url: <?php echo admin_url('admin-ajax.php'); ?>,
			type: 'GET',
			dataType: "json",
			contentType: "application/json; charset=utf-8",
			cache: false,
			data: {
				'action': 'get_post_detail'
			},
			success: function (data, textStatus, xhr) {
				$("#post").html(data.data.content);
			},
			error: function (xhr, status, error) {
				alert(xhr.responseJSON.html);
			}
		});
});
```

چه زمانی باید از admin-ajax استفاده کنیم؟

از روش ایجاد درخواست های ایجکس در وردپرس بوسیله ی admin-ajax تنها برای مدیریت آن هم درخواست هایی که پروسس زیادی ندارند میبایست استفاده شود.استفاده از روش Admin-ajax برای قالب و افزونه ها در بخش فرانت اند کاملا غیر استاندارد می باشد.

ایجکس در وردپرس بر اساس متد وب سرویس یا REST API

بیایید مثال قبل را به روش REST API وردپرس انجام دهیم.مزیت استفاده از روش REST API وردپرس نسبت به admin-ajax در دو چیز می باشد:

۱. اگر در حال برنامه نویسی یک استارتاپ هستید که در پلتفرم های مختلفی قرار هست آن را پیاده سازی کنید. شما با این کار، یک بار برنامه نویسی می کنید و در همه جا استفاده می کنید.

۲. سرعت وب سرویس وردپرس در تست های انجام شده ۱۵٪ بیش تر از درخواست های admin-ajaxوردپرس می باشد.

دقت کنید :در وب سرویس یا REST API وردپرس دیگر احراز هویت و Cookie ها فعال نیستن به همین دلیل شما می بایست تابع قوانین Permission Callback در REST API وردپرس باشید.

برای ایجاد یک Route جدید در وب سرویس وردپرس از تابع register_rest_route استفاده می کنیم:

```
# Create WordPress REST API Route
add_action('rest_api_init', 'create_route');
function create_route()
{
	register_rest_route('post', 'get_detail', [
		'methods' => 'GET',
		'callback' => 'get_post_detail'
	]);
}

# Get List Post Method
function get_post_detail(WP_REST_Request $request)
{
	$post = get_post(1);
	$array = [
		'title' => $post->post_title,
		'link' => get_the_permalink(1),
		'content' => $post->post_content
	];

	return new WP_REST_Response($array, 200);



}
```

حال اگر در مرورگر آدرس زیر را وارد کنید، خروجی وب سرویس وردپرس را مشاهده خواهید کرد:

```
http://site.com/wp-json/post/get_detail
```

اگر با خطای ۴۰۴ وردپرس مواجه شده اید ، نیازمند آن هستید که یکبار پیوندهای یکتای وردپرس را بروز رسانی یا Flush کنید.ساده ترین راه برای این کار استفاده از **خط فرمان وردپرس** (WP-CLI)و دستور زیر می باشد:

```
wp rewrite flush
```

ایجکس در وردپرس بر اساس متد استانداردRewrite API

برای ایجاد درخواست های ایجکس بر اساس کلاس Rewrite API در قدم اول میبایست یک آدرس جدید در وردپرس معرفی کنیم. برای اینکار ساده ترین روش استفاده از تابع add_rewrite_rule میباشد.من قبلا مطلبی برای مثال های کاربردی تر این تابع زدم که میتوانی از این جا مطالعه کنی:

حتما بخوانید : افزایش امنیت درخواست GET در برنامه نویسی استارتاپ وردپرس

مسیر جدید را می توانیم در هوک وردپرس init قرار دهیم به سادگی:

```
# Define New Rewrite Url Params
add_action('init', 'define_ajax_url');
function define_ajax_url()
{
	add_rewrite_tag('%rewrite_ajax%', '([^&/]+)');
	add_rewrite_rule('ajax/?([^/]*)',
'index.php?rewrite_ajax=$matches[1]', 'top');
}
```

همیشه بعد از اینکه تغییراتی در Rewrite وردپرس ایجاد شد ، باید یکبار پیوند های یکتا را در بخش تنظیمات وردپرس بروز رسانی کنید. ما توسط کد بالا به وردپرس یک آدرس جدید معرفی کردیم.

اگر میخواهید در افزونه های وردپرس از Rewrite API استفاده کنید مثلا یک مسیر ایجکس جدید معرفی کنید یا یک پست تایپ ایجاد کنید ، حتما میبایست در قسمت register_activation_hook یکبار عملیات Flush را اجرا کنید.

حال میتوانیم دستورات خود را براحتی برنامه نویسی کنیم:

```
# Define Get Post Detail
add_action('template_redirect', 'get_post_detail');
function get_post_detail()
{
	global $wp_query;

	# Don't do anything unless this is an AJAX request
	$method = $wp_query->get('rewrite_ajax');
	if (!$method) {
		return;
	}

	# Check method name
	if ($method == "get_post_detail") {
		$post = get_post(1);
		$array = [
			'title' => $post->post_title,
			'link' => get_the_permalink(1),
			'content' => $post->post_content
		];

		wp_send_json_success($array, 200);
	}


}
```

در کد بالا ابتدا بررسی کردیم ، آیا Rewrite Tag که معرفی کردیم به نام rewrite_ajax در آدرس درخواستی کاربر وجود دارد یا خیر. در صورت وجود داشتن و برابر بودن آن مقدار با get_post_detail کد Json برای کاربر به نمایش در میاید. شما میتوانید دستورات متعدد زیادی بوسیله همین شرط آن جا اضافه کنید.

اگر کاربر وارد آدرس زیر در مرورگر شود خروجی را خواهد دید:

```
http://site.com/ajax/get_post_detail
```

اگر دقت کرده باشید در روش Rewrite API حتی نوع آدرس درخواست ها هم به انتخاب خودمان هست و کاملا آزاد هستیم.
خروجی کار به فرمت Json بدین شکل هست:

```
{
    "success": true,
    "data": {
        "title": "سلام دنیا!",
        "link": "http://site.com/سلام-دنیا/",
        "content": "به وردپرس خوش آمدید. این اولین نوشتهٔ شماست!"
    }
}
```

در Rewrite API هم مشابه Admin Ajax ما به کوکی های وردپرس دسترسی داریم این بدان معناست که توابع احراز هویت بدون هیچ تنظیمات خاصی اجرا می شوند.اگر به درخواست های ایجکس افزونه ووکامرس دقت کرده باشید همه ی آن ها در یک مسیر جداگانه ای با پسوند wc-ajax می باشند. این بدین معنا هست که افزونه ووکامرس نیز از Rewrite API استفاده کرده است.

آقای درزی سوال پیش اومده واسه خیلی ها تو این لحظه، اگر متد Rewrite API سرعت آن از REST APIو Admin Ajax بیش تر است پس چرا اصلا REST API بوجود آمد؟ خب همه ی توسعه دهندگان وردپرس ی سره از این استفاده می کردند؟

پاسخ این سوال واضح هست ، ما در وب سرویس یا REST API وردپرس، استاندارد هایی داریم که برای ارتباط بین سرور و پلتفرم ها از نظر پروتکل HTTP و امنیتی الزامی هستند مثل بحث authentication.پس نباید چنین مقایسه ای انجام داد و هر کدام را در جای خودش میبایست استفاده کرد.

نتیجه گیری ایجکس در وردپرس

نتیجه گیری بدین شکل شد ، شما برای ایجکس در افزونه و قالب در بخش فرانت اند و سایت از استفاده REST API استفاده می کنید. برای اپلیکیشن ها و پلتفرم های مختلف از Rewrite API هم تنها برای محدود درخواست های داخل مدیریت بهره مند می شوید Admin Ajax می کنید. از.