Towards the WordPress RESTful API - Integrated and Collaborative Communication

Presentation · October 2019

DOI: 10.13140/RG.2.2.11255.57767

2 authors:

Shafiq Lutaaya
Makerere University
35 PUBLICATIONS **2** CITATIONS

SEE PROFILE

Hive Collab
Makerere University
2 PUBLICATIONS **0** CITATIONS

SEE PROFILE

WORDPRESS KAMPALA MEETUP

Towards the WordPress RESTful API

INTEGRATED AND COLLABORATIVE COMMUNICATION

Shafiq Lutaaya,

Ronzag.com

E: lutayashafiqholmes@gmail.com

P:+256702772721

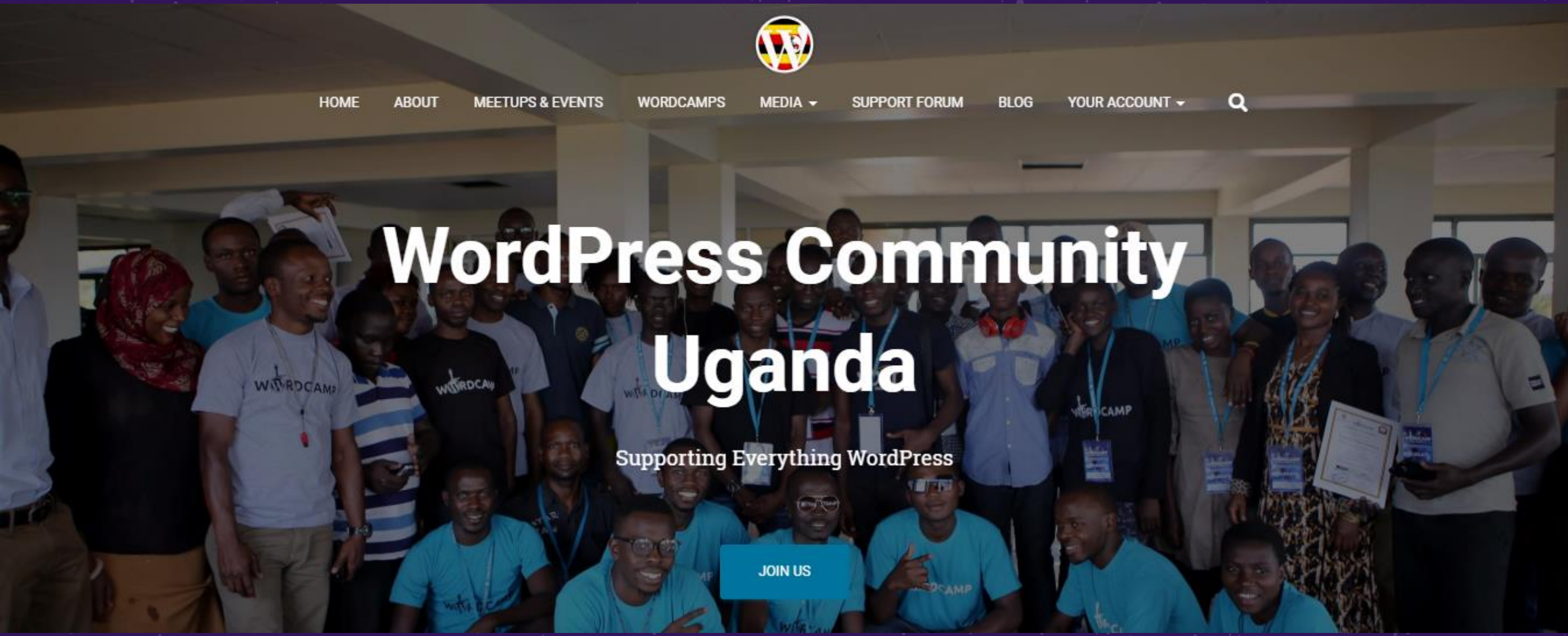
12th October 2019

Hive Collab,

Kanjokya House,

Kanjokya Street, Kamwokya, Kampala, UG

The WordPress Community in Uganda: wcuganda.org



Objectives

- Introduction to API technology
- Web Services: (XML, JSON based)
- Enterprise Integration and Collaborative Communication
- SOAP Vs REST
- What is SOAP ,WSDL

- The WordPress RESTful API
  - Why use the WordPress RESTful API
  - Key Concepts (Routes and Endpoints, Requests, Responses, Schema, Controller Classes)
    1. Routes and Endpoints,
    2. Requests,
    3. Responses,
    4. Schema,
    5. Controller Classes

The WordPress RESTful API Handbook

demo.wp-api.org/wp-json

- REST API Handbook
- Reference
- Using the REST API
- Extending the REST API
- Changelog

Other WordPress APIs

WordPress APIs

Languages: **English** • Русский • API 日本語 (Add your language)

The **WordPress API** stands for the WordPress Application Programming Interface. It can be separated into multiple API sections / topics. Each covers the functions involved in and use of a given set of functionality. Together they form what might be called the WordPress API, which is the plugin/theme/add-on interface created by the entire WordPress project.

If you've read through all of these you should have a good sense of how to extend WordPress through Plugins.

- Dashboard Widgets API
- Database API
- HTTP API
- REST API
- File Header API
- Filesystem API
- Metadata API
- Options API
- Plugin API
- Quicktags API
- Rewrite API
- Settings API
- Shortcode API
- Theme Modification API
- Theme Customization API
- Transients API
- Widgets API
- XML-RPC WordPress API (supersedes the legacy Blogger, MovableType, and metaWeblog APIs)

See also WordPress.org API

What is an API? Why do we care about them?

- An API is an Application Program Interface
- Consider a soda machine
  - It has an interface
  - Users authenticate and submit requests through the interface
  - The machine sends users responses
- An API is a fancy word for “making it easier for this bit of software to work with that other bit of software so that they can both do stuff together”

There are several APIs in WordPress that you probably already use (whether you realize it or not)

- **Database API**: Allows you to interact with the wp_options table in MySQL by way of *add_option()*. Also deals with transients and metadata in WordPress.
- **Shortcode API**: Allows you to register WordPress shortcodes via *add_shortcode()*
- **Plugin API**: when you install a plugin, that plugin "hooks" into the rest of WordPress via the Plugin API
- **Rewrite API**: Manages writing your WordPress URLs for you. When you go to *Settings —> Permalinks* in your dashboard and specify a permalink pattern for your pages and posts, those settings are applied to WordPress by way of the Rewrite API
- **HTTP API**: Allows you to have WordPress submit HTTP requests to other servers (for example, when you upgrade WordPress from your admin area, WordPress is submitting an HTTP request to WordPress.org by way of the HTTP API)

There are about 15 APIs in WordPress today. What's so special about the WP REST API?

- The APIs that we've talked about have to do with **WordPress interacting with WordPress**. For example:
  - Having WordPress add a setting to the WordPress database in MySQL
  - Adding a WordPress plugin to your WordPress website

    Registering a WordPress plugin with your WordPress website

    Having WordPress rewrite WordPress's urls

REST-ful APIs for the rest of us

- REST = **RE**presentative **S**tate **T**ransfer
- It's not a specific tool or framework; it's more just a way of thinking about things
- There are a handful of specific properties an API must have in order to be considered RESTful; for our purposes today, the most important of these properties is ***statelessness***
- Statelessness boils down to the idea that everything a server needs in order to handle a request *must be contained within the request itself.* Put another way, the server is not allowed to share information between requests (this is why browser cookies are a thing, if you've ever wondered)
- Because the server can't share information between requests, every resource (for example, a page, a post, etc) you want to expose through your API must have a unique Uniform Resource Locator that can be specified in an individual request (a URL for example, or a URL in combination with some query strings)

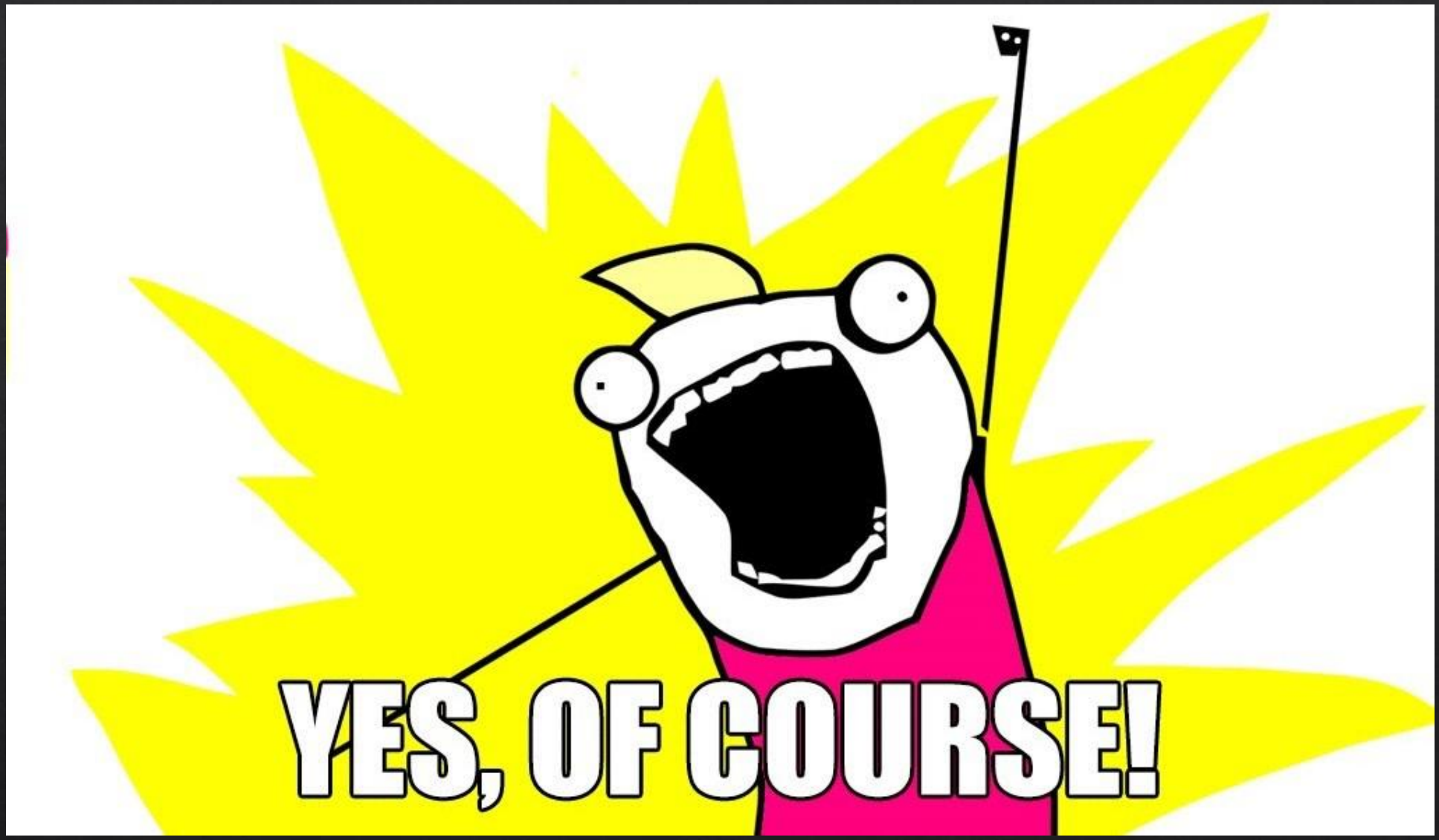
The WordPress REST API: Who cares if it's easier to work with?

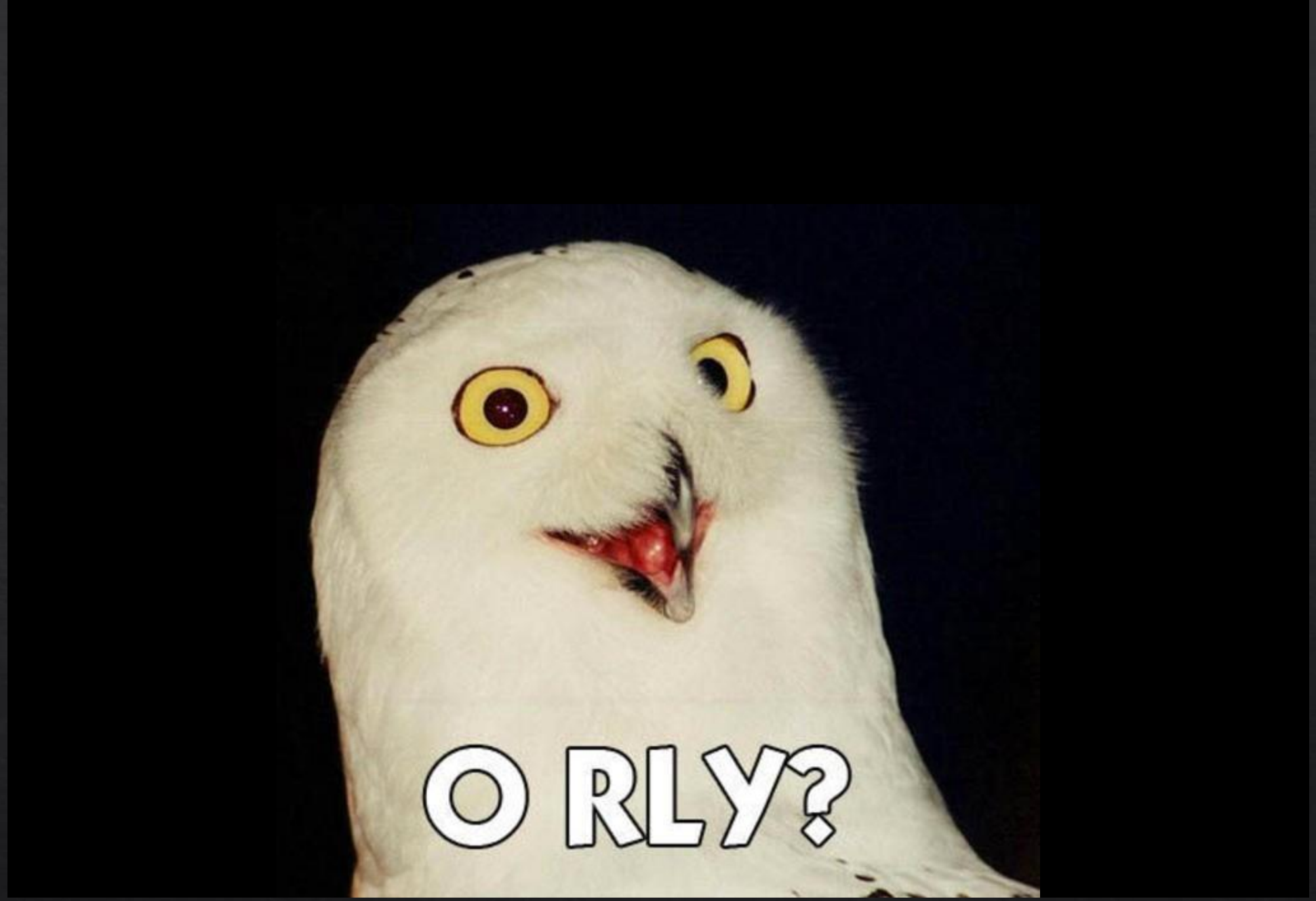
- Regardless of which API is "better"
  - REST and JSON are widely understood both by programming languages and by programmers (and even, arguably, non-programmers)
  - HTTP, too, is well understood and widely supported
  - XML… not as much
  - WordPress's XML-RPC API requires developers to know something about WordPress and the XML-RPC protocol in order to use it
  - Depending on what software you are using, XML-RPC support may be poor
  - It is very likely to be more complex dealing with XML-RPC than it would be dealing with HTTP requests and JSON, no matter what language you're using (don't take my word for it: try submitting an XML-RPC request to WordPress, then try submitting one through the REST API)

WordPress as a web application

If you come across something that you can't do with WordPress, or that you could do much better with some other software, the WordPress REST API makes integrating that other software with WordPress - and finding developers who can do that work for you - much much easier

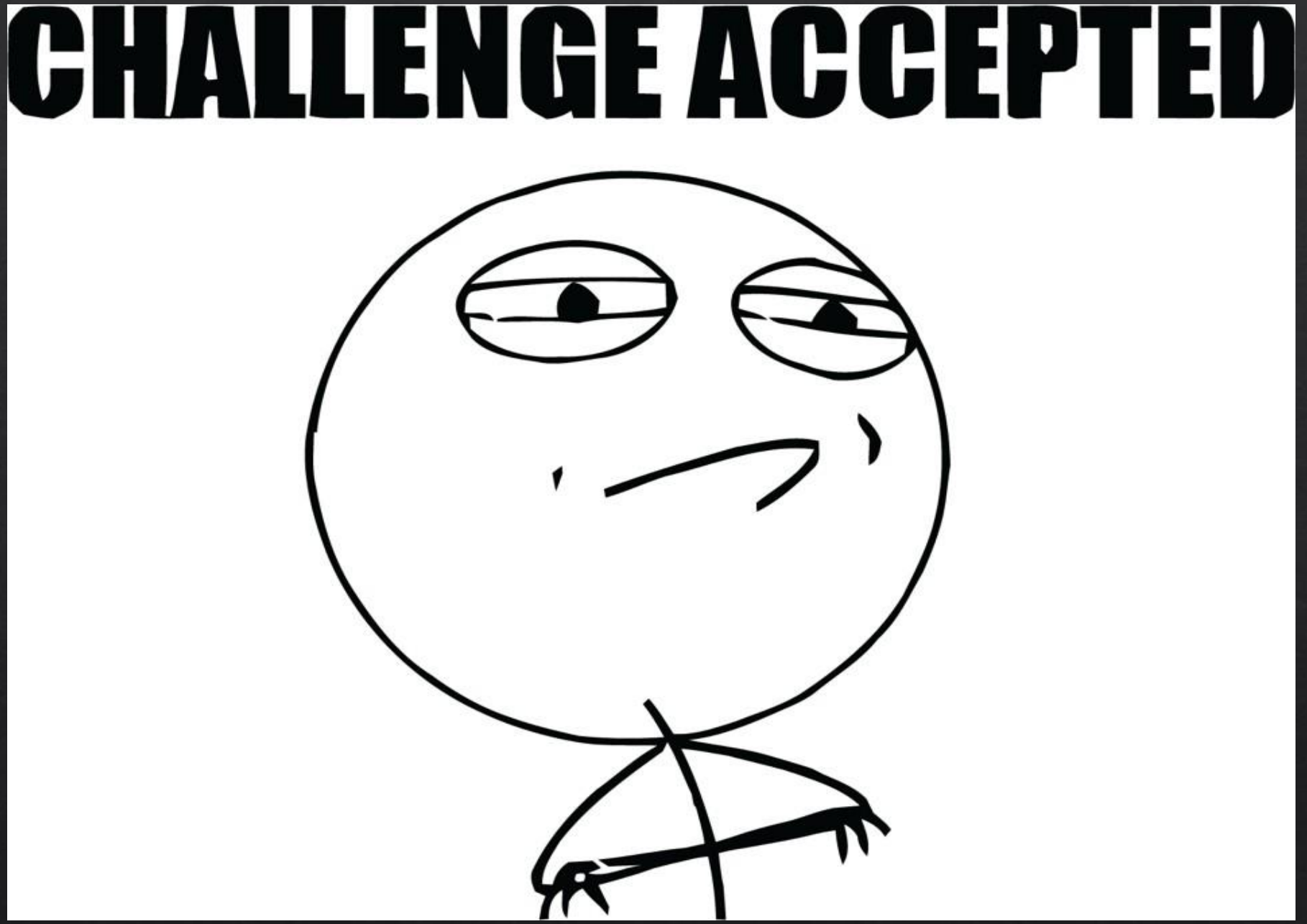
Can WordPress be the best choice for next project?


YES, OF COURSE!


O RLY?


OH, SO WORDPRESS CAN BE USED
FOR EVERY KIND OF PROJECT?
TELL ME ABOUT ANGULAR, REACT JS,
MOBILE APPS, MICROSERVICES, ...

WordPress could not be the best solution


CHALLENGE ACCEPTED

WP REST API

WP REST API

WP REST API

WP REST API

The methods

***GET*:** With this method, you can fetch information from the server.

***POST*:** This enables you to send information to the server in question.

***PUT*:** With the *put* method, you can edit and update existing data.

***DELETE*:** This enables you to delete information.

VERSION 1

```
GET /posts
```

VERSION 2

```
GET /wp-json/wp/v2/posts
```

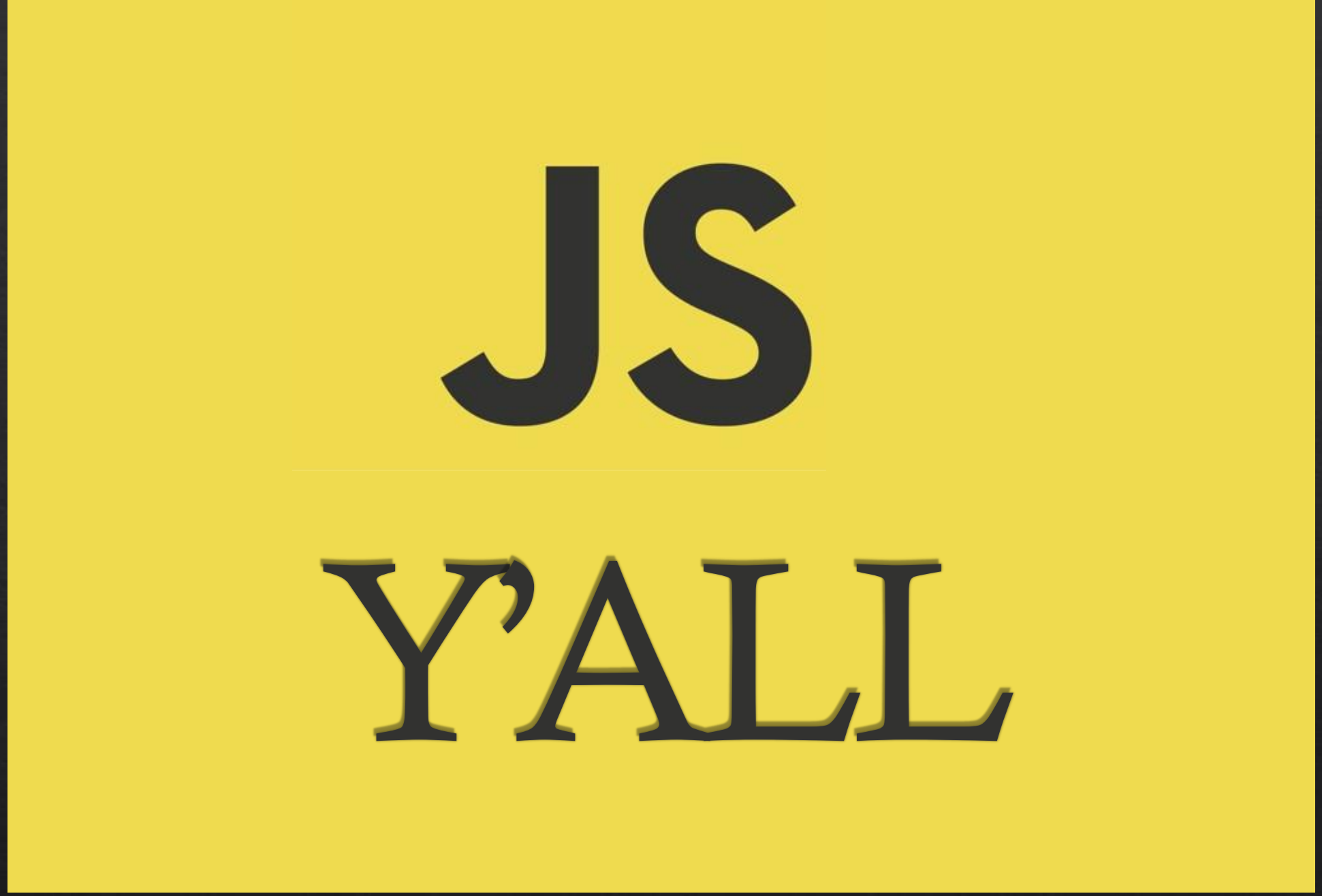
- **Endpoints available in the plugin**

WP REST API


```
{
    "name": "My WordPress Site",
    "description": "Just another WordPress site",  "URL":
    "http:\/\/example.com",
    "routes": {
            "\/": {
                    "supports": [
                            "HEAD",
                            "GET"
                    ],
                    "meta": {
                        "self": "http:\/\/example.com\/wp-json\/"
                    }
            }
    ...
```

Why WP REST API matters?


WHAT IF I TOLD YOU
PHP ISN'T THE ONLY ONE


JS
Y'ALL

MOBILE APPS


SKY IS NOT THE LIMIT
YOUR IMAGINATION IS

WP REST API REFERENCE

/wp-json/wp/v2	Routes and endpoints available
/wp-json/wp/v2/posts	Create, read, update and delete posts
/wp-json/wp/v2/pages	Create, read, update and delete pages
/wp-json/wp/v2/users	Create, read, update and delete users
/wp-json/wp/v2/media	Create, read, update and delete media items
/wp-json/wp/v2/taxonomies	Read taxonomies and terms

GET /wp-json/wp/v2/posts	List posts
GET /wp-json/wp/v2/posts/<id>	Read the post <id>
POST /wp-json/wp/v2/posts	Create a new post
PUT /wp-json/wp/v2/posts/<id>	Update the post <id>
DELETE /wp-json/wp/v2/posts/<id>	Delete the post <id>

Extending WP REST API

Extending

- We are able to access posts, pages etc.
- What about Custom post types and custom fields?

CUSTOM FIELDS

WP REST API

```php
add_action( 'rest_api_init', 'slug_register_starship' );
function slug_register_starship() {
    register_api_field( 'post',
        'starship',
        array(
            'get_callback'    => 'slug_get_starship',
            'update_callback' => null,
            'schema'          => null,
        )
    );
}

function slug_get_starship( $object, $field_name, $request ) {
    return get_post_meta( $object[ 'id' ], $field_name, true );
}
```

WP REST API

```
  type: "post",
  link: "http://vagrant.local/post-1101/",
- title: {
      rendered: "Post 1101"
  },
- content: {
      rendered: "<p>This is the post content</p> "
  },
- excerpt: {
      rendered: "<p>This is the post content</p> "
  },
  author: 1,
  featured_image: 0,
  comment_status: "open",
  ping_status: "open",
  sticky: false,
  format: "standard",
  starship: "Millennium Falcon",
```

ACF to WP-API

Plugs Advanced Custom Fields (ACF) data into the WordPress JSON API (WP-API).

Download Version 1.3.2

Description | Installation | FAQ | Screenshots | Changelog | Stats | Support | Reviews | Developers

Puts all ACF fields from posts, pages, custom post types, comments, attachments and taxonomy terms, into the WP-API output under the 'acf' key. Creates a new `/option` endpoint returning options (requires ACF Options Page plugin).

Requires: 3.9.0 or higher
Compatible up to: 4.3.0
Last Updated: 2 months ago
Active Installs: 900+

https://wordpress.org/plugins/acf-to-wp-api/

WP REST API


```
  sticky: false,
  format: "standard",
- acf: {
      acf_field: "This is the content of the ACF Field",
    - acf_image: {
          id: 1107,
          alt: "",
          title: "e64c7d89f26bd1972efa854d13d7dd61",
          caption: "",
          description: "",
          mime_type: "image/jpeg",
          url: "http://vagrant.local/content/uploads/2015/10/e64c7d89f26bd1972efa854d13d7dd61.jpg",
          width: 48,
          height: 48,
        - sizes: {
              thumbnail: "http://vagrant.local/content/uploads/2015/10/e64c7d89f26bd1972efa854d13d7dd61.jpg",
              thumbnail-width: 48,
              thumbnail-height: 48,
              medium: "http://vagrant.local/content/uploads/2015/10/e64c7d89f26bd1972efa854d13d7dd61.jpg",
              medium-width: 48,
              medium-height: 48,
              large: "http://vagrant.local/content/uploads/2015/10/e64c7d89f26bd1972efa854d13d7dd61.jpg",
              large-width: 48,
              large-height: 48,
              post-thumbnail: "http://vagrant.local/content/uploads/2015/10/e64c7d89f26bd1972efa854d13d7dd61.jpg",
              post-thumbnail-width: 48,
              post-thumbnail-height: 48
          }
      }
  },
```

CUSTOM POST TYPES


```php
$args = array(
    'labels'             => $labels,
    'description'        => __( 'Description.', 'your-plugin-textdomain' ),
    'rewrite'            => array( 'slug' => 'book' ),
    'capability_type'    => 'post',
    //...
    'show_in_rest'       => true,
    'rest_base'          => 'books',
    'rest_controller_class' => 'WP_REST_Posts_Controller',
    'supports'           => array( 'title', 'editor',
                                   'author', 'thumbnail',
                                   'excerpt', 'comments' )
);

register_post_type( 'book', $args );
```

GET /wp-json/wp/v2/books/<id>	Read the book <id>

CONCLUSIONS… FOR REAL

WP REST API allow developers to build faster and reactive applications

THANKS

www.ronzag.com

QUESTIONS

REFERENCE

https://developer.wordpress.com/docs/api/

https://make.wordpress.org/core/2015/10/28/rest-api-welcome-the-infrastructure-to-core/

http://jacklenox.com/2015/03/30/building-themes-with-the-wp-rest-api-wordcamp-london-march-2015/

https://github.com/kadamwhite/wordpress-rest-api

https://github.com/WP-API/client-js

The End