Hardcore URL Routing for WordPress

WordCamp Atlanta - March 15th, 2014

Mike Schinkel
@mikeschinkel
about.me/mikeschinkel

To Cover

- URLs We Love
- How Do URLs Get Routed?
- How to see your Rewrite Rules
- Review the Rewrite API Functions & Hooks
- Ways to Add Rewrite Rules
- flush_rewrite_rules( ) & generate_rewrite_rules()
- Full-control Manual URL Routing
- Link Generation: Beyond Routing
- Other Stuff we Won't Cover

URLs We Love

- **`/products/{product_name}/details/`**
  - Post type => `'product'`
- **`/products/{prod_cat}/{product_name}/`**
  - Post type => `'product'`, Taxonomy (`'prod_cat'`) `term`
- **`/{permalink_path}/json/`**
  - A JSON version of your post
- **`/comments/{year}/{monthnum}/{day}/`**
  - Archive of all comments not related to a specific post
- **`/api/{api_request}/`**
  - “Virtual” URL for Custom code

More URLs We Love

- **`/{automaker}/`**
  - Post type => `'automaker'`
- **`/{automaker}/{model}/`**
  - Post type => `'automaker'`, Post type => `'model'`
- **`/{user_nicename}/`**
  - Specified user
- **`Other?`**
  - Taking requests....

How Do URLs Get Routed?

- WP Matches URL Path to a *Rewrite Pattern*
  - Rewrite Patterns are *Regular Expressions*
- Example URL Path:
  - **/2014/03/15/hello-world/**
- Matching Regular Expression
  - **([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)(/[0-9]+)?/?$**
- Rewrite Pattern:
  - **index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&page=$matches[5]**
- Resultant Query String:
  - **index.php?year=2014&monthnum=03&day=15&name=hello-world&page=**

`index.php` in Rewrite Pattern?

See: **`.htaccess`**:*

```
# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
</IfModule>
# END WordPress
```

* *If on IIS then* ***`web.config`*** *which has a different format.*

The Resultant Query String

- Stored in **`$wp->query_vars`**
- Processed by **`WP_Query()`** for The Loop

```
// Not exactly this, but similar
$query_vars = 'year=2014&monthnum=03&day=15&name=hello-world&page=';
$query= new WP_Query( $query_vars );
```

- Essentially the same as this:

```
$query = new WP_Query( array(
  'year'     => 2014,
  'monthnum' => 3,
  'day'      => 15,
  'name'     => 'hello-world',
  'page'     => '',
));
```

See your Rewrite Rules

```
<?php
/*
 * Filename: /rewrite-rules.php
 * REMEMBER TO SET YOUR PERMALINKS! Settings > Permalinks
 */

include __DIR__ . '/wp-load.php';
$rewrite_rules = get_option( 'rewrite_rules' );
header( 'Content-type: text/plain;' );
print_r( $rewrite_rules );
```

Rewrite Rules, in a Table!

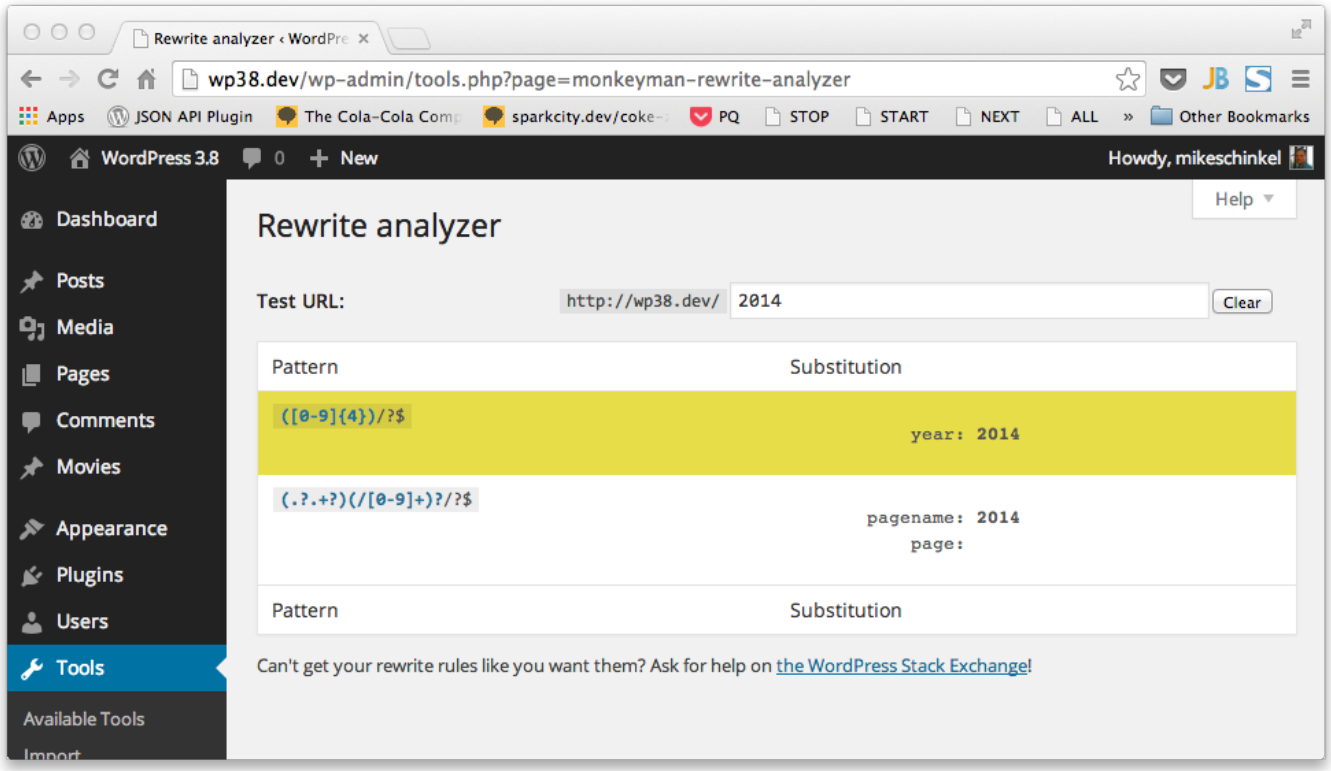
```
<?php
/*
 * Filename: /rewrite-rules2.php
 * REMEMBER TO SET YOUR PERMALINKS! Settings > Permalinks
 */

include __DIR__ . '/wp-load.php';
$rewrite_rules = get_option( 'rewrite_rules' );
?>

<html>
<head>
  <title>Rewrite Rules</title>
  <style type="text/css"> th { text-align: left; } </style>
</head>
<body>
  <table>
    <?php
    foreach( $rewrite_rules as $regex => $pattern ) {
      echo "<tr><th>{$regex}</th><td>{$pattern}</td></tr>";
    } ?>
  </table>
</body>
</html>
```

Best Way to See Your Rulez

- Monkeyman Rewrite Analyzer
  - http://wordpress.org/plugins/monkeyman-rewrite-analyzer/



The WordPress Rewrite API

- Rewrite API Functions
  - Related Functions
- Rewrite API Hooks
  - Related Hooks

The Rewrite API Functions

- **`register_post_type()*`** and **`register_taxonomy()*`**
- **`flush_rewrite_rules()`**
- **`add_rewrite_rule()`** and **`add_rewrite_tag()`**
- **`get_query_var()*`**
- **`register_activation_hook()`** and **`register_deactivation_hook()`**

******Related functions*

The Rewrite API Hooks

- `'parse_request'`
- `'request'`
- `'after_switch_theme'*`
- `'template_redirect'*`
- `'template_include'*`
- `'redirect_canonical'*`
- `'do_parse_request'`
- `'query_var'`

**Related Hooks*

Different Ways to Add Rules

- Adding Rewrite Rules Indirectly
- Adding Rewrite Tags & Rules
- Validating Rewritten URLs
- Adding EndPoints
- Adding Permastructs
- Adding External Rules

Adding Rules Indirectly

/products/{product_name}/

```
<?php // functions.php

add_action( 'init', 'x_init' );
function x_init() {
  register_post_type( 'x_product', array(
    'public' => true,
    'label' => __( 'Products', 'X' ),
    'rewrite' => array(
      'with_front' => false,
      'slug' => 'products',
    ),
  ));
}
```

Always Remember to Flush!

`flush_rewrite_rules( $hard_or_soft )`

```
add_action( 'init', 'x_init' );
function x_init() {

  // Call register_post_type() here

  if ( WP_DEBUG ) {
   if ( ! function_exists( 'save_mod_rewrite_rules' ) ) {
     require_once( ABSPATH . 'wp-admin/includes/file.php' );
     require_once( ABSPATH . 'wp-admin/includes/misc.php' );
   }
   flush_rewrite_rules( true );
  }
}
```

- **`true`** (hard) - Writes changes to **`.htaccess/web.config`**
- **`false`** (soft) - Does not writes changes.

Flushing in a Plugin

```php
<?php
/**
 * Plugin Name: Hardcore URLs
 */
class Hardcore_Urls {
  static function on_load() {
    add_action( 'init', array( __CLASS__, '_init' ) );
    register_activation_hook( __FILE__, array( __CLASS__, '_activate' ) );
    register_deactivation_hook( __FILE__, array( __CLASS__, '_deactivate' ) );
  }
  static function _init() {
    // Register post types, taxonomies here
    // Also add rewrite rules, tags, permastructs, etc. here.
  }
  static function _activate() {
    flush_rewrite_rules( true );
  }
  static function _deactivate() {
    flush_rewrite_rules( true );
  }
}
Hardcore_Urls::on_load();
```

Flushing in a Theme

```
<?php // functions.php

class Hardcore_Urls_Theme {
  static function on_load() {
    add_action( 'init', array( __CLASS__, '_init' ) );
    add_action( 'after_switch_theme', '_after_switch_theme' );
  }
  static function _init() {
    // Register post types, taxonomies here
    // Also add rewrite rules, tags, permastructs, etc. here.
  }
  static function _after_switch_theme() {
    flush_rewrite_rules( true );
  }
}
Hardcore_Urls_Theme::on_load();
```

Adding Tags and Rewrite Rules

/products/{product_name}/details/

```
register_post_type( 'x_product', array(
  'public' => true,
  'label' => __( 'Products', 'X' ),
  'query_var' => 'x_product',
  'rewrite' => array(
    'with_front' => false,
    'slug' => 'products',
  ),
));
add_rewrite_tag( '%x_product_details%', 'details' );
add_rewrite_rule(
  'products/([^/]+)/details/?$',
  'index.php?x_product=$matches[1]&' .
            'x_product_details=true',
  'top'
);
```

Using in a Theme

Example: `single-acw14_product.php`

```
<?php get_header(); ?>
  <div id="primary" class="content-area">
    <div id="content" class="site-content" role="main">
      <?php
        if ( get_query_var( 'x_product_details' ) ) {
          echo '<h2>THIS IS A PRODUCT DETAILS PAGE</h2>';
        }
      ?>
      <?php
        while ( have_posts() ) : the_post();
          get_template_part( 'content', get_post_format() );
          if ( comments_open() || get_comments_number() ) {
            comments_template();
          }
        endwhile;
      ?>
    </div>
  </div>
<?php get_footer();
```

Add Taxonomy Term to Post Type Slug

/products/{prod_cat}/{product_name}/

```
register_post_type( 'x_product', array(
  'public' => true,
  'label' => __( 'Products', 'X' ),
  'rewrite' => array(
    'with_front' => false,
    'slug' => 'products/[^/]+',
  ),
));
```

Except! No `{prod_cat}` Validation:

/products/valid-cat/{product_name}/
/products/foo-bar/{product_name}/

Validating {prod_cat}

/products/{prod_cat}/{product_name}/

```
add_action( 'init', 'x_init' );
function x_init() {
  // First register post type and taxonomy here
  add_rewrite_tag( '%x_prod_cat%', '([^/]+)' );
  add_rewrite_rule(
    'products/([^/]+)/([^/]+)/?$',
    'index.php?x_prod_cat=$matches[1]&x_product=$matches[2]',
    'top'
  );
}
add_filter( 'parse_request', 'x_parse_request' );
function x_parse_request( $wp ) {
  if ( ! empty( $wp->query_vars['x_prod_cat'] ) ) {
    $term_slug = $wp->query_vars['x_prod_cat'];
    $query = new WP_Query( $wp->query_vars );
    if ( ! has_term( $term_slug, 'x_prod_cat', $query->post ) ) {
      $wp->query_vars = array(
        'error'     => 404,
        'post_type' => true
      );
      status_header( 404 );
      nocache_headers();
    }
  }
}
```

More Rewrite Functions

- `add_rewrite_endpoint()`
- `add_permastruct()`
- `WP_Rewrite` Class
  - `$wp_rewrite->add_external_rule()`
- `generate_rewrite_rules()` *
- `save_mod_rewrite_rules()` and `iis7_save_url_rewrite_rules()`
- `get_sample_permalink()` *
- `redirect_guess_404_permalink()!!!`

**Called internally by WordPress*
!!! Bad Actor!

Adding an EndPoint

/products/{product_name}/json/

```
add_filter( 'request', 'x_request' );
function x_request( $query_vars ) {
  if( isset( $query_vars['json'] ) ) {
    $query_vars['json'] = true;
  }
  return $query_vars;
}
add_filter( 'template_redirect', 'x_template_redirect' );
function x_template_redirect() {
  if( get_query_var( 'json' ) ) {
    global $post;
    header( 'Content-type: application/json' );
    echo json_encode( $post );
    exit;
  }
}
```

Adding a Permastruct

/comments/{year}/{monthnum}/{day}/

```
add_action( 'init', 'x_init' );
function x_init() {
  // Register other things here
  add_permastruct(
    'x_comments',
    'comments/%year%/%monthnum%/%day%',
    array(
      'with_front'  => true,
      'ep_mask'     => EP_NONE,
      'paged'       => true,
      'feed'        => true,
      'forcomments' => false,
      'walk_dirs'   => true,
      'endpoints'   => true,
    )
  );
  // Flush rules if WP_DEBUG
}
```

Recognizing a Permastruct

/comments/{year}/{monthnum}/{day}/

```
add_filter( 'parse_request', 'x_parse_request' );
function x_parse_request( $wp ) {
  $regex = '#^comments/?([0-9]{4}/?([0-9]{1,2}/?([0-9]{1,2}/?(.*))?)?)?$#';
  if ( preg_match( $regex, $wp->request, $match ) ) {
    $wp->query_vars += array(
      'post_type' => true,
      'x_comments_permastruct' => true
    );
  }
}
```

** Why Yes! That is a rather ugly regular expression!*

Preparing a Special Template

/comments/{year}/{monthnum}/{day}/

```
add_filter( 'template_include', 'x_template_include' );
function x_template_include( $template_file ) {
  global $wp, $wp_the_query;
  if( get_query_var( 'x_comments_permastruct' ) ) {
    $wp_the_query->queried_object_id = null;
    $wp_the_query->queried_object = $query = new WP_Comment_Query;
    $date_query = array();
    foreach( $wp->query_vars as $var_name => $var_value ) {
      if ( preg_match( '#^(year|monthnum|day)$#', $var_name ) ) {
        $date_query[] = array( $var_name => $var_value );
      }
    }
    $query->comments = $query->query( array(
      'date_query' => $date_query,
    ));
    x_load_comments( get_stylesheet_directory() . '/comment.php' );
  }
  return $template_file;
}
```

Serving a Special Template

/comments/{year}/{monthnum}/{day}/

```
function x_load_comments( $_template_file ) {
  global $post, $wp_rewrite, $wpdb, $wp_version, $wp, $user_ID;
  global $comments, $comment, $wp_comment_query;
  $wp_comment_query = get_queried_object();
  $comments = $wp_comment_query->comments;
  $comment = reset( $comments );
  $post = get_post( $comment->comment_post_ID );
  require( $_template_file );
  exit;
}
```

A Special Comment Template

/comments/{year}/{monthnum}/{day}/

```
<?php // comment.php
get_header(); ?>
<div id="main-content" class="main-content">
  <div id="primary" class="content-area">
    <div id="content" class="site-content" role="main">
    <?php if ( count( $comments ) ) :
        echo '<ul>';
        foreach( $comments as $comment ):
          $post = get_post( $comment->comment_post_ID );
          $url = get_comment_link( $comment );
          $excerpt = get_comment_excerpt( $comment );
          $date_time = get_comment_date() . ' '. get_comment_time();
          echo "<li>About: "; the_title();
          echo "<br>{$comment->comment_author} &lt;{$date_time}&gt;";
          echo "<br><a href=\"{$url}\">{$excerpt}</a><hr>";
        endforeach;
        echo '</ul>';
      else:
        get_template_part( 'content', 'none' );
      endif; ?>
    </div><!-- #content -->
  </div><!-- #primary -->
  <?php get_sidebar( 'content' ); ?>
</div><!-- #main-content --><?php
get_sidebar();
get_footer();
```

Rules Generated by Permastruct

/comments/{year}/{monthnum}/{day}/

```
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]&withcomments=1
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]&withcomments=1
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/page/?([0-9]{1,})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&paged=$matches[4]
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&cpage=$matches[4]
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/json(/(.*))?/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&json=$matches[5]
comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]
comments/([0-9]{4})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]&withcomments=1
comments/([0-9]{4})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]&withcomments=1
comments/([0-9]{4})/([0-9]{1,2})/page/?([0-9]{1,})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&paged=$matches[3]
comments/([0-9]{4})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&cpage=$matches[3]
comments/([0-9]{4})/([0-9]{1,2})/json(/(.*))?/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]&json=$matches[4]
comments/([0-9]{4})/([0-9]{1,2})/?$
=> index.php?year=$matches[1]&monthnum=$matches[2]
comments/([0-9]{4})/feed/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&feed=$matches[2]&withcomments=1
comments/([0-9]{4})/(feed|rdf|rss|rss2|atom)/?$
=> index.php?year=$matches[1]&feed=$matches[2]&withcomments=1
comments/([0-9]{4})/page/?([0-9]{1,})/?$
=> index.php?year=$matches[1]&paged=$matches[2]
comments/([0-9]{4})/comment-page-([0-9]{1,})/?$
=> index.php?year=$matches[1]&cpage=$matches[2]
comments/([0-9]{4})/json(/(.*))?/?$
=> index.php?year=$matches[1]&json=$matches[3]
comments/([0-9]{4})/?$
=> index.php?year=$matches[1]
```

This was for all `true` & `'ep_mask' => EP_ALL`

Adding an External Rule

/api/{whatevah}/

```
add_action( 'init', 'x_init' );
function x_init() {
  global $wp_rewrite;
  $local_path = substr( __DIR__ . '/api.php', strlen( ABSPATH ) );
  $wp_rewrite->add_external_rule( 'api/?.*$', $local_path );
}
```

Adds the following to **`.htaccess`**:

```
RewriteRule ^api/?.*$ /wp-content/themes/wp38/api.php [QSA,L]
```

Recognizing the External Rule

/api/{whatevah}/

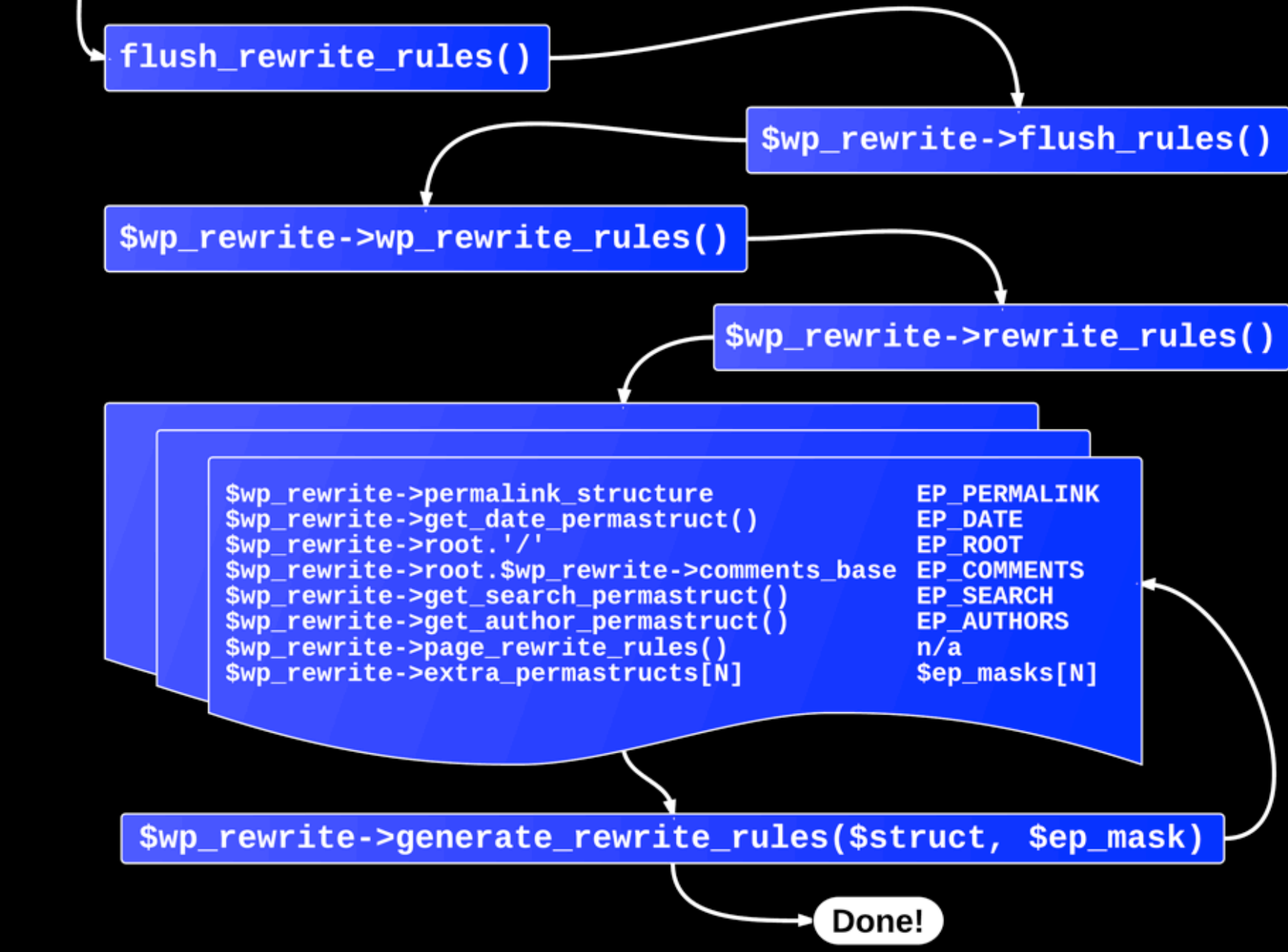
```
<?php
/* Filename: api.php
 * This is only if you need WordPress loaded.
 * The "/../../.." assumes in a plugin directory.
 */
require( __DIR__ . '/../../../wp-load.php' );

$url_path = $_SERVER['REQUEST_URI'];
$request = preg_replace( '#^/api/(.*)$#', '$1',
$url_path );
echo "Your API request was: {$request}";
```

The `generate_rewrite_rules()` Function

- This is where it gets **really** ugly.
- This can turn your brain to mush.
- You might even *switch to Drupal!*
- Not really. Who'd want Drupal ***if they can haz*** WordPress?!?
- But I digress...

Regenerating Rewrite Rules



Hooks Used During Rule Generation

- 'delete_option_rewrite_rules'
- 'pre_get_option_rewrite_rules'
- 'default_option_rewrite_rules'
- 'post_rewrite_rules'
- 'date_rewrite_rules'
- 'root_rewrite_rules'
- 'comments_rewrite_rules'
- 'search_rewrite_rules'
- 'author_rewrite_rules'
- 'page_rewrite_rules'
- "{$permastructname}_rewrite_rules"
- 'tag_rewrite_rules'
- 'generate_rewrite_rules'
- 'rewrite_rules_array'

For 'post_rewrite_rules'

```
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/attachment/([^/]+)/?$] => index.php?attachment=$matches[1]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/attachment/([^/]+)/trackback/?$] => index.php?attachment=$matches[1]&tb=1
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/attachment/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?attachment=$matches[1]&feed=$matches[2]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/attachment/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?attachment=$matches[1]&feed=$matches[2]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/attachment/([^/]+)/comment-page-([0-9]{1,})/?$] => index.php?attachment=$matches[1]&cpage=$matches[2]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/trackback/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&tb=1
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&feed=$matches[5]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&feed=$matches[5]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&paged=$matches[5]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&cpage=$matches[5]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)/json(/(.*))?/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&json=$matches[6]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/([^/]+)(/[0-9]+)?/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&name=$matches[4]&page=$matches[5]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/([^/]+)/?$] => index.php?attachment=$matches[1]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/([^/]+)/trackback/?$] => index.php?attachment=$matches[1]&tb=1
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?attachment=$matches[1]&feed=$matches[2]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?attachment=$matches[1]&feed=$matches[2]
[[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}/[^/]+/([^/]+)/comment-page-([0-9]{1,})/?$] => index.php?attachment=$matches[1]&cpage=$matches[2]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&paged=$matches[4]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&cpage=$matches[4]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/json(/(.*))?/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&json=$matches[5]
[([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]
[([0-9]{4})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]
[([0-9]{4})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]
[([0-9]{4})/([0-9]{1,2})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&paged=$matches[3]
[([0-9]{4})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&cpage=$matches[3]
[([0-9]{4})/([0-9]{1,2})/json(/(.*))?/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&json=$matches[4]
[([0-9]{4})/([0-9]{1,2})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]
[([0-9]{4})/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]
[([0-9]{4})/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]
[([0-9]{4})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&paged=$matches[2]
[([0-9]{4})/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&cpage=$matches[2]
[([0-9]{4})/json(/(.*))?/?$] => index.php?year=$matches[1]&json=$matches[3]
[([0-9]{4})/?$] => index.php?year=$matches[1]
```

For 'date_rewrite_rules'

```
  [([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]
  [([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]
  [([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/page/?([0-9]{1,})/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&paged=$matches[4]
  [([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]
  [([0-9]{4})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]
  [([0-9]{4})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]
  [([0-9]{4})/([0-9]{1,2})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&paged=$matches[3]
  [([0-9]{4})/([0-9]{1,2})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]
  [([0-9]{4})/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]
  [([0-9]{4})/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]
  [([0-9]{4})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&paged=$matches[2]
  [([0-9]{4})/?$] => index.php?year=$matches[1]
```

For 'root_rewrite_rules'

```
[feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?&feed=$matches[1]
[(feed|rdf|rss|rss2|atom)/?$] => index.php?&feed=$matches[1]
[page/?([0-9]{1,})/?$] => index.php?&paged=$matches[1]
```

For 'comments_rewrite_rules'

```
[comments/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?&feed=$matches[1]&withcomments=1
[comments/(feed|rdf|rss|rss2|atom)/?$] => index.php?&feed=$matches[1]&withcomments=1
```

For 'search_rewrite_rules'

```
[search/(.+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?s=$matches[1]&feed=$matches[2]
[search/(.+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?s=$matches[1]&feed=$matches[2]
[search/(.+)/page/?([0-9]{1,})/?$] => index.php?s=$matches[1]&paged=$matches[2]
[search/(.+)/?$] => index.php?s=$matches[1]
```

For 'author_rewrite_rules'

```
[author/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?author_name=$matches[1]&feed=$matches[2]
[author/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?author_name=$matches[1]&feed=$matches[2]
[author/([^/]+)/page/?([0-9]{1,})/?$] => index.php?author_name=$matches[1]&paged=$matches[2]
[author/([^/]+)/?$] => index.php?author_name=$matches[1]
```

For 'page_rewrite_rules'

```
  [.?.+?/attachment/([^/]+)/?$] => index.php?attachment=$matches[1]
  [.?.+?/attachment/([^/]+)/trackback/?$] => index.php?attachment=$matches[1]&tb=1
  [.?.+?/attachment/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?attachment=$matches[1]&feed=$matches[2]
  [.?.+?/attachment/([^/]+)/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?attachment=$matches[1]&feed=$matches[2]
  [.?.+?/attachment/([^/]+)/comment-page-([0-9]{1,})/?$]
=> index.php?attachment=$matches[1]&cpage=$matches[2]
  [(.?.+?)/trackback/?$] => index.php?pagename=$matches[1]&tb=1
  [(.?.+?)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?pagename=$matches[1]&feed=$matches[2]
  [(.?.+?)/(feed|rdf|rss|rss2|atom)/?$] => index.php?pagename=$matches[1]&feed=$matches[2]
  [(.?.+?)/page/?([0-9]{1,})/?$] => index.php?pagename=$matches[1]&paged=$matches[2]
  [(.?.+?)/comment-page-([0-9]{1,})/?$] => index.php?pagename=$matches[1]&cpage=$matches[2]
  [(.?.+?)(/[0-9]+)?/?$] => index.php?pagename=$matches[1]&page=$matches[2]
```

For 'category_rewrite_rules'

```
[category/(.+?)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?category_name=$matches[1]&feed=$matches[2]
[category/(.+?)/(feed|rdf|rss|rss2|atom)/?$] => index.php?category_name=$matches[1]&feed=$matches[2]
[category/(.+?)/page/?([0-9]{1,})/?$] => index.php?category_name=$matches[1]&paged=$matches[2]
[category/(.+?)/?$] => index.php?category_name=$matches[1]
```

For 'post_tag_rewrite_rules'

```
[tag/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?tag=$matches[1]&feed=$matches[2]
[tag/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?tag=$matches[1]&feed=$matches[2]
[tag/([^/]+)/page/?([0-9]{1,})/?$] => index.php?tag=$matches[1]&paged=$matches[2]
[tag/([^/]+)/?$] => index.php?tag=$matches[1]
```

For 'tag_rewrite_rules'

```
[tag/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?tag=$matches[1]&feed=$matches[2]
[tag/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?tag=$matches[1]&feed=$matches[2]
[tag/([^/]+)/page/?([0-9]{1,})/?$] => index.php?tag=$matches[1]&paged=$matches[2]
[tag/([^/]+)/?$] => index.php?tag=$matches[1]
```

For 'post_format_rewrite_rules'

```
[type/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?post_format=$matches[1]&feed=$matches[2]
[type/([^/]+)/(feed|rdf|rss|rss2|atom)/?$] => index.php?post_format=$matches[1]&feed=$matches[2]
[type/([^/]+)/page/?([0-9]{1,})/?$] => index.php?post_format=$matches[1]&paged=$matches[2]
[type/([^/]+)/?$] => index.php?post_format=$matches[1]
```

For 'x_comments_rewrite_rules'

```
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]&withcomments=1
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&feed=$matches[4]&withcomments=1
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/page/?([0-9]{1,})/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&paged=$matches[4]
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&cpage=$matches[4]
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/json(/(.*))?/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]&json=$matches[5]
  [comments/([0-9]{4})/([0-9]{1,2})/([0-9]{1,2})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&day=$matches[3]
  [comments/([0-9]{4})/([0-9]{1,2})/feed/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]&withcomments=1
  [comments/([0-9]{4})/([0-9]{1,2})/(feed|rdf|rss|rss2|atom)/?$]
=> index.php?year=$matches[1]&monthnum=$matches[2]&feed=$matches[3]&withcomments=1
  [comments/([0-9]{4})/([0-9]{1,2})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&paged=$matches[3]
  [comments/([0-9]{4})/([0-9]{1,2})/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&cpage=$matches[3]
  [comments/([0-9]{4})/([0-9]{1,2})/json(/(.*))?/?$] => index.php?year=$matches[1]&monthnum=$matches[2]&json=$matches[4]
  [comments/([0-9]{4})/([0-9]{1,2})/?$] => index.php?year=$matches[1]&monthnum=$matches[2]
  [comments/([0-9]{4})/feed/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]&withcomments=1
  [comments/([0-9]{4})/(feed|rdf|rss|rss2|atom)/?$] => index.php?year=$matches[1]&feed=$matches[2]&withcomments=1
  [comments/([0-9]{4})/page/?([0-9]{1,})/?$] => index.php?year=$matches[1]&paged=$matches[2]
  [comments/([0-9]{4})/comment-page-([0-9]{1,})/?$] => index.php?year=$matches[1]&cpage=$matches[2]
  [comments/([0-9]{4})/json(/(.*))?/?$] => index.php?year=$matches[1]&json=$matches[3]
  [comments/([0-9]{4})/?$] => index.php?year=$matches[1]
```

Whew!

- I hope you never have to use any of that!
- But if you do, you are now equipped.
- *(Assumed your brain didn't turn to mush.)*

The `'do_parse_request'` Hook

- This is the **strong** magic.
- Using `'do_parse_request'` is like....
- ...getting the **Glengarry Leads**!!!
- Seriously! It's **omni-**powerful!
- But Theme/Plugin Devs *beware*...
- The magic might *overpower* you.

Context Specific URLs

- WordPress only supports *Context Free* URLs
- But we often want *Context Sensitive* URLs
- `'do_parse_request'` makes it possible
- But, it's not without risk
- Plugins and Themes won't expect it
- And you can have one URL hide another
- *With Great Power Comes Great Responsibility*

Post Type Name in Root

/{product_name}/

```
add_filter( 'do_parse_request', 'x_do_parse_request', 10, 3 );
function x_do_parse_request( $continue, $wp, $extra_query_vars ) {
  if ( $continue ) {
    $url_path = trim( $_SERVER['REQUEST_URI'], '/' );
    $url_path = sanitize_title_with_dashes( $url_path );
    $query = new WP_Query( array(
      'name'        => $url_path,
      'post_type'   => 'x_product',
      'post_status' => 'publish',
    ));
    if ( 1 == $query->found_posts ) {
      $wp->query_vars = array(
        'post_type' => 'x_product',
        'x_product' => $url_path,
        'name'      => $url_path,
        'page'      => '',
      );
      $continue = false;  // IMPORTANT
    }
  }
  return $continue;
}
```

Parent and Child Post Types

Part 1 of 3

/{product_name}/{model_name}/

```
add_filter( 'do_parse_request', 'x_do_parse_request', 10, 3 );
function x_do_parse_request( $continue, $wp, $extra_query_vars ) {
  if ( $continue && ! is_admin() ) {
    $url_path = trim( $_SERVER['REQUEST_URI'], '/' );
    if ( ! empty( $url_path ) && '?' != $url_path[0] ) {
      $url_path_parts = array_map( 'sanitize_title_with_dashes', explode( '/',
$url_path ) );
      switch( count( $url_path_parts ) ) {
        case 2:
          if ( $model = x_match_post_name( 'x_model', $model_name = $url_path_parts[1] ) ) {
            if ( $product = x_match_post_name( 'x_product', $url_path_parts[0] ) ) {
              if ( $model->post_parent == $product->ID ) {
                x_route_post_name( $wp, 'x_model', $model_name );
                $continue = false;
              }
            }
          }
          break;
        case 1:
          $continue = ! x_match_post_name( 'x_product', $url_path_parts[0], $wp );
          break;
      }
    }
  }
  return $continue;
}
```

Parent and Child Post Types

Part 2 of 3

/{product_name}/{model_name}/

```
function x_route_post_name( $post_type, $post_name, $wp ) {
  $wp->query_vars = array(
    'post_type' => $post_type,
    $post_type  => $post_name,
    'name'      => $post_name,
    'page'      => '',
  );
}
function x_match_post_name( $post_type, $post_name, $wp = false ) {
  $query = new WP_Query( array(
    'name'        => $post_name,
    'post_type'   => $post_type,
    'post_status' => 'publish',
  ));
  if ( ( $matched = 1 == $query->found_posts ) && $wp ) {
    x_route_post_name( $wp, $post_type, $post_name );
  }
  return $matched ? $query->post : false;
}
```

Parent and Child Post Types

Part 3 of 3

/{product_name}/{model_name}/

```
add_action( 'add_meta_boxes', 'x_add_meta_boxes' );
function x_add_meta_boxes( $post_type ) {
  if ( 'x_model' == $post_type ) {
    add_meta_box( 'product_parent_box', // $id
      __( 'Parent Product', 'X' ),      // $title
      'x_model_parent_metabox',         // $callback
      'x_model',                        // $post_type
      'side',                           // $context
      'low'                             // $priority
    );
  }
}
function x_model_parent_metabox( $post ) {
  $post_type_object = get_post_type_object( 'x_product' );
  $post_type_object->hierarchical = true;
  wp_dropdown_pages( array(
    'name' => 'parent_id',
    'post_type' => 'x_product',
    'selected' => $post->post_parent,
    'show_option_none' => __( 'Select a Parent', 'X' ),
    'option_none_value' => 0,
  ));
  $post_type_object->hierarchical = false;
}
```

Stopping the "*Helpful*" Guesses

/{product_name}/{model_name}/
=>
?post_type=x_model&p={post_id}

```
add_action( 'redirect_canonical', 'x_redirect_canonical', 10, 2 );
function x_redirect_canonical( $redirect_url, $requested_url ) {
  list( $url, $query ) = explode( '?', "{$redirect_url}?" );
  if ( $url == home_url('/') ) {
    parse_str( $query, $params );
    if ( isset( $params['post_type'] ) && 'x_model' == $params['post_type'] ) {
      // Dammit; redirect_guess_404_permalink() interfered! Set it back!!!
      $redirect_url = $requested_url;
    }
  }
  return $redirect_url;
}
```

Reveal the Query Vars

```
add_action( 'shutdown', 'x_shutdown' );
function x_shutdown() {
  if ( ! is_admin() ) {
    global $wp;
    echo '<pre><code><font size="7">';
    echo '$wp->query_vars: ';
    print_r( $wp->query_vars );
    echo '</font></code></pre>';
  }
}
```

Performance Optimization

- Cache Top *N* URLs in `wp_options`

  In array; key=URL, value=`$wp->query_vars`

- Periodically Recalculate Top *N*
- Even Better if using Memcache
- This is your Homework!

Warning:
May Be Incompatible

- Highly Custom Sites:
  - A-OK
- Normal Websites or Blog:
  - Plugins and themes might be incompatible
- Plugins or Themes for Distribution:
  - Be very careful; this is non-standard routing
- Also...
  - Very easy to create ambiguity in routing

The Dispatch Library

- Using URI Templates as Patterns
- Inspecting URLs by Segment
- Plan to do lots of caching
- Still PRE-ALPHA
- `github.com/newclarity/dispatch`

But Routing Ain't Enough

- Link Generation
- Allow Editing the Slug
- Changing the Permalink Sample

Generate the Link

/{product_name}/{model_name}/

```
add_filter( 'post_type_link', 'x_post_type_link', 10, 2 );
function x_post_type_link( $post_link, $post ) {
  switch ( $post->post_type ) {
    case 'x_product':
      $post_link = home_url( "/{$post->post_name}/" );
      break;
    case 'x_model':
      $product_link = x_post_type_link( false, get_post( $post->post_parent ) );
      $post_link = "{$product_link}{$post->post_name}/";
      break;
  }
  return $post_link;
}
```

Make the Slug Editable

/{product_name}/ {model_name} /

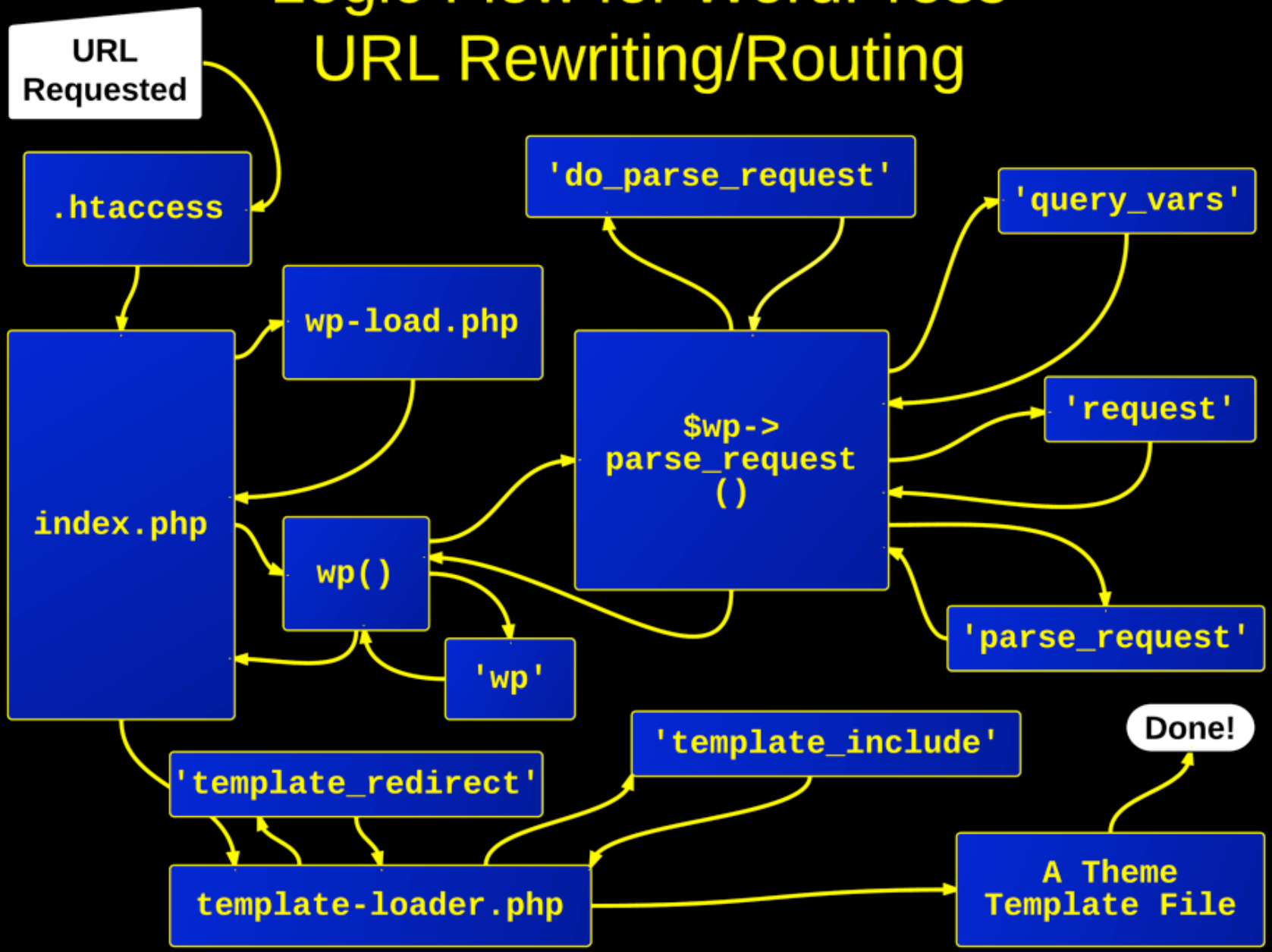
```
add_filter( 'post_type_link', 'x_post_type_link', 10, 3 );
function x_post_type_link( $post_link, $post, $leavename = false ) {
  $slug = $leavename ? "%{$post->post_type}%" : $post->post_name;
  switch ( $post->post_type ) {
    case 'x_product':
      $post_link = home_url( "/{$slug}/" );
      break;
    case 'x_model':
      $product_link = x_post_type_link( false, get_post( $post->post_parent ) );
      $post_link = "{$product_link}{$slug}/";
      break;
  }
  return $post_link;
}
```

Changing the Sample Permalink

----->[{model_name}]<-----

```
add_filter( 'post_type_link', 'x_post_type_link', 10, 3 );
function x_post_type_link( $post_link, $post, $leavename = false ) {
  $slug = $leavename ? "%{$post->post_type}%" : $post->post_name;
  switch ( $post->post_type ) {
    case 'x_product':
      $post_link = home_url( "/{$slug}/" );
      break;
    case 'x_model':
      $product_link = x_post_type_link( false, get_post( $post->post_parent ) );
      $post_link = "{$product_link}{$slug}/";
      break;
  }
  if ( $sample ) {
    // NO CLUE WHY YOU'D WANT TO DO THIS... BUT YOU CAN!
    $post_link = "----->[ %{$post->post_type}% ]<-----";
  }
  return $post_link;
}
```

Logic Flow for WordPress URL Rewriting/Routing



In Review

What We Learned Today

- How WordPress Routes URLs
- `index.php` and `.htaccess`
- That URLs ultimately create `$args` for `WP_Query()`
- Rewrite Rules and how to see/test them.
- How and why to Flush Rules
- The Rewrite API Functions
  - Especially `add_rewrite_rule()` and `add_rewrite_tag()`
- The Rewrite API Hooks
  - Especially `'parse_request'`and `'do_parse_request'`

In Review *(cont'd)*

What Else We Learned Today

- About Endpoints, Permastructs and External Rules
- How abysmal `generate_rewrite_rules()` is
- All them crazii hooks when generating.
- The **Strong Magic**: do_parse_request
  - Provides *Context Sensitive URLs*
  - How to discover the query vars needed to route URLs
- Generating links, editing elugs and changing samples.
- And finally the logic flow for URL rewriting/routing.

About the 'x_' Prefix

See: http://nacin.com/2010/05/11/in-wordpress-prefix-everything/

*I just used 'x_' (**FOR THIS TALK, ONLY**) because it was short!*

And Stuff I Wrote for the Intro

- But I decided it didn't flow.
- Like how scenes gets left...
- on *the cutting room floor*.
- So I added to the DVD. ;-)

WordPress' URL Patterns

- Post & Page URL Patterns
- Archive URL Patterns
- Feed URL Patterns
- Other URL Patterns

Post & Page URL Patterns

Posts	`/{year}/{month}/{day}/{postname}/`
Pages	`/{pagename}/`
Hierarchical Pages	`/{rootname}/../{leafname}/`
Custom Post Types	`/{post_type_slug}/{postname}/`
Attachments	`/{y}/{m}/{d}/attachment/{attachment}/`
	`/{whatever}/attachment/{attachment}/`
	`/{whatever}/{attachment}/`

Archive URL Patterns

Tag Archive	`/tag/{tag}/`
Category Archive	`/category/{category}/`
Author Archive	`/author/{author_name}/`
Search Results	`/search/{s}/` and `/s?={s}`
Post Format Archive	`/type/{post_format}/`
Custom Post Type Archive	`/{post_type_slug}/`

Also Day, Month and Year Archives.

Feed URL Patterns

Feed*	`/feed/{feed|rdf|rss|rss2|atom}/`
Category Feed*	`/category/{category}/feed/{feed_type}/`
Tag Feed*	`/tag/{tag}/feed/{feed_type}/`
Comment Feed*	`/comment/feed/{feed_type}/`
Search Results Feed*	`/search/{s}/feed/{feed_type}/`
Author Feed*	`/author/{author_name}/feed/{feed_type}/`
* {feed_type} =	**feed, rss, rdf, rss2, atom**

Also Attachment, Post, Page, Day, Month and Year Feeds. (Why?)

Other URL Patterns

Robots	`/robots.txt`
Register	`/{whatever}/wp-register.php`
Comments	`/{post_url}/comment-page-{cpage}/`
Trackback	`/{post_url}/trackback/{whatever}`
And some...	`I'm sure I forgot about.`

What is URL Routing?

- *The mapping of a URL Path of an HTTP Request identifying the Resource on your WordPress site for which you want an HTTP Response to return a Representation.*
- Whew! What a mouthful!

Why Do We Care?

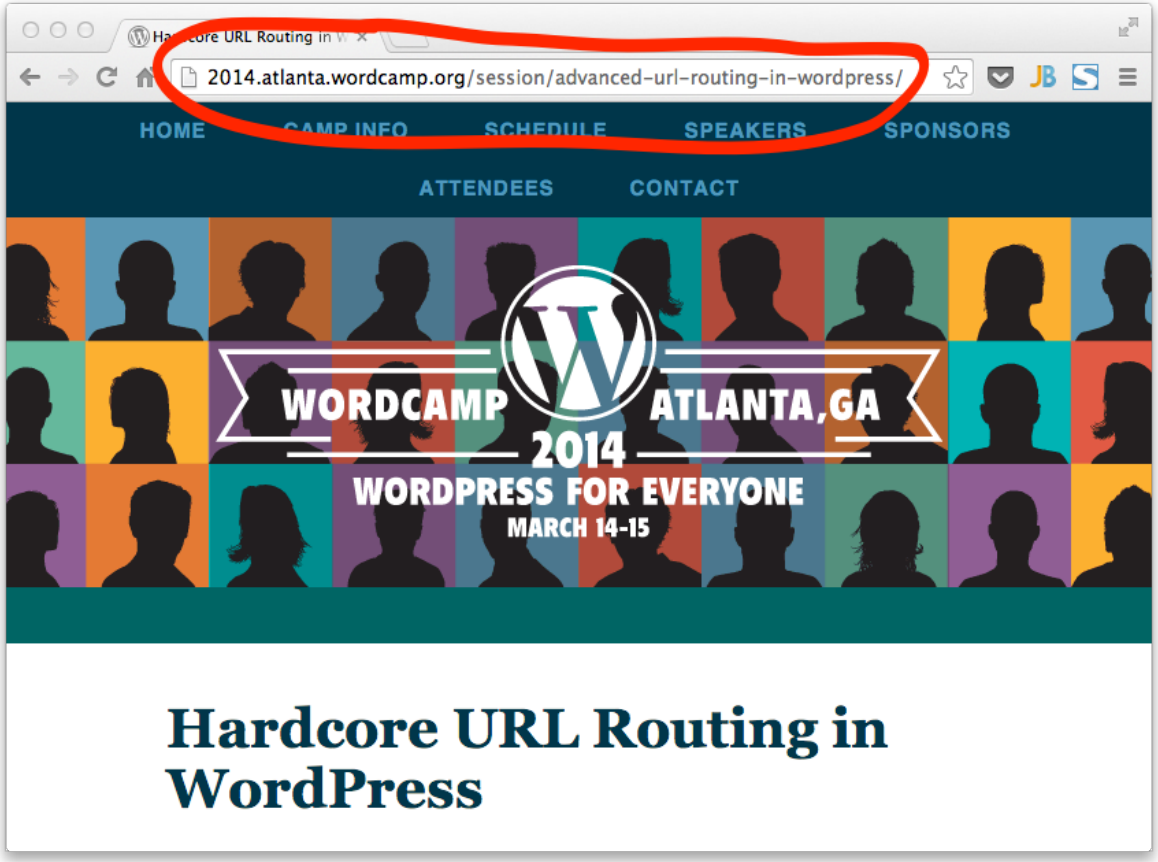
- Because we want to control what *Representation* is returned for a given *URL Path*.
- So let's define those terms...

What is a *URL Path*?

- Example URL:
  - **http://example.com/2014/03/15/hello-world/**
- The URL Path is:
  - **/2014/03/15/hello-world/**

What is an *HTTP Request?*

It's when you type a URL into your browser and press Enter!



What is a *Resource?*

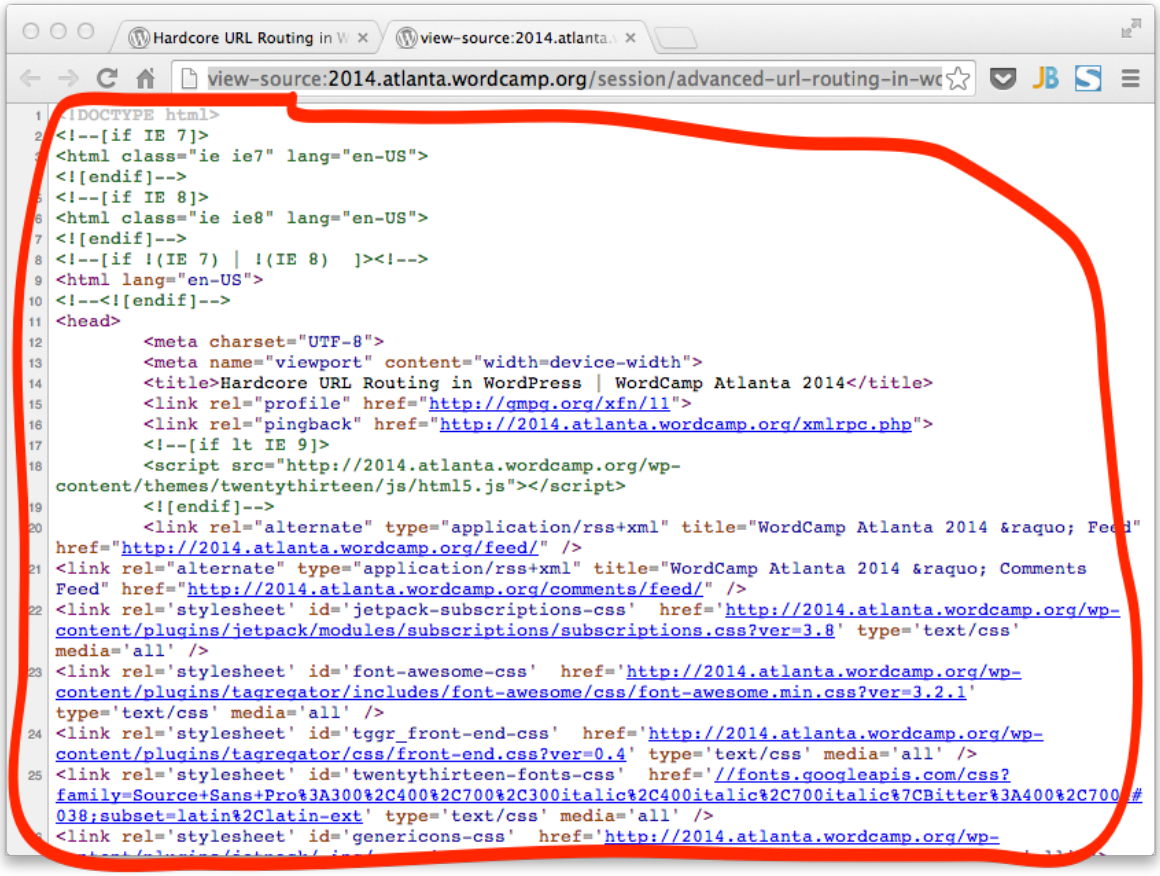
- It's the ***idea*** of the thing you want, the thing located at the URL you type into your browser.
- Don't worry, nerd types have argued over this definition for years!
- A Representation is simply the HTML file, image or other file you want when you enter a URL.

What is an *HTTP Response?*

- It is what your WordPress website packages up and sends back to your browser and is based on the URL path you request.

What is a *Representation?*

It's what gets downloaded when you request a URL with your browser.



See, That Wasn't Hard!

- So in review:

  URL Routing is the mapping of a URL Path of an HTTP Request identifying the Resource on your WordPress site for which you want an HTTP Response to return a Representation.

Future Reading

- The Rewrite API: The Basics
  - http://code.tutsplus.com/tutorials/the-rewrite-api-the-basics--wp-25474
- The Rewrite API: Post Types & Taxonomies
  - http://code.tutsplus.com/articles/the-rewrite-api-post-types-taxonomies--wp-25488
- A (Mostly) Complete Guide to the WordPress Rewrite API
  - http://o.pmg.co/a-mostly-complete-guide-to-the-wordpress-rewrite-api
- Some background on rewrite rules
  - http://wordpress.stackexchange.com/a/5478/89

Me, in More Detail

- **President**, NewClarity LLC
  - **Boutique** WordPress consulting firm with a small but **expert team**.
- Specialize in **complex** sites & vertical market **products**
  - **Offering:** Back-end architecture, implementation, ***training***, code reviews.
- **375+ Answers** at WordPress Answers
  - Was a founding moderator
- Have blogged at **HardcoreWP.com**
  - Hope to find time again *(**doh!**)*
- Several ***"Libraries"*** *for* WordPress *at github.com/newclarity*
  - **Exo** - "*MVC that doesn't change WordPress, it just makes it stronger.*"
  - **Sunrise** - *Code-based Forms and Fields*
  - **Dispatch** - *Hardcore URL Routing for WordPress*
- **Contact us** for more of the ***"Strong Magic"***
  - mike@newclarity.net

Thanks for Attending

May all your URLs be *pretty*
and may they all *match*
their intended patterns

Mike Schinkel

@mikeschinkel

mike@newclarity.net