

10 YEARS OF COMMUNITY

Welcome to WordCamp Toronto

Network: RU-Secure

Username: trsguest2017

Password: RUguest253\$



CANADA 150

HTML in Functions Kills Kittens

MVC In WordPress Plugins and Themes

Matt Graham - @themattyg
WordCamp Toronto 2017



I am @themattyg

Resume (Cole's/Cliff's Notes Version)

- Graduated from Durham College Multimedia Design 2009
- Full Stack PHP/JS Developer for 10 years
- WordPress Developer for 5 years, almost exclusively for 2 years
- Worked with Retail, Ad Agencies, Restaurants, and Education as employee and contractor
- Jack of all digital trades: Audio & Video editor, Live Sound, Photography, Video & Audio Solutions





**I DOAN
WANTS 2 DYE**

WordPress is not MVC

That's OK. We aren't talking about Core.

```

function import_options() {
    $j = 0;

7>
<form action="php echo admin_url( 'admin.php?import=wordpress&amp;step=2' ); ?" method="post">
    <?php wp_nonce_field( 'action: 'import=wordpress' ); ?>
    <input type="hidden" name="import_id" value="php echo $this-&gt;id; ?" />

<?php if ( ! empty( $this->authors ) ) : ?>
    <h3><?php _e( 'Assign Authors', 'domain: 'wordpress-importer' ); ?></h3>
    <p><?php _e( 'To make it easier for you to edit and save the imported content, you may want to reassign the author of the imported item to an existing user of' ); ?>
<?php if ( $this->allow_create_users() ) : ?>
    <p><?php printf( ___, 'If a new user is created by WordPress, a new password will be randomly generated and the new user's role will be set as %s. Manually change the role to %s.' ); ?>
<?php endif; ?>
    <ol id="authors">
<?php foreach ( $this->authors as $author ) : ?>
        <li><?php $this->author_select( $j++, $author ); ?></li>
<?php endforeach; ?>
    </ol>
<?php endif; ?>

<?php if ( $this->allow_fetch_attachments() ) : ?>
    <h3><?php _e( 'Import Attachments', 'domain: 'wordpress-importer' ); ?></h3>
    <p>
        <input type="checkbox" value="1" name="fetch_attachments" id="import-attachments" />
        <label for="import-attachments"><?php _e( 'Download and import file attachments', 'domain: 'wordpress-importer' ); ?></label>
    </p>
<?php endif; ?>

    <p class="submit"><input type="submit" class="button" value="php esc_attr_e( 'Submit', 'domain: 'wordpress-importer' ); ?" /></p>
</form>
<?php
)

/++

```

```

public static function gform_standard_settings($position, $form_id) {
    if ($position == 1680) {
        echo "<li class='repeater_settings field_setting'>";
        <label for='field_repeater_start'>Start ";

        gform_tooltip( name: 'form_field_repeater_start');

        echo " </label>";
        <input type='number' id='field_repeater_start' min='1' value='1' onchange='SetFieldProperty('start', this.value);'>
        </li>";

        echo "<li class='repeater_settings field_setting'>";
        <label for='field_repeater_min'>Min ";

        gform_tooltip( name: 'form_field_repeater_min');

        echo " </label>";
        <input type='number' id='field_repeater_min' min='1' value='1' onchange='SetFieldProperty('min', this.value);'>
        </li>";

        echo "<li class='repeater_settings field_setting'>";
        <label for='field_repeater_max'>Max ";

        gform_tooltip( name: 'form_field_repeater_max');

        echo " </label>";
        <input type='number' id='field_repeater_max' min='1' onchange='SetFieldProperty('max', this.value);'>
        </li>";
    }
}

public static function gform_appearance_settings($position, $form_id) {
    if ($position == 408) {
        echo "<li class='repeater_settings field_setting'>";
        <input type='checkbox' id='field_repeater_hideLabel' onchange='SetFieldProperty('hideLabel', this.checked);'>
        <label for='field_repeater_hideLabel' class='inline'>Hide Label & Description ";

        gform_tooltip( name: 'form_field_repeater_hideLabel');

        echo " </label>";
        </li>";
    }
}

```

```
public static function gform_standard_settings($position, $form_id) {
    if ($position == 1680) {
        echo "<li class='repeater_settings field_setting'>
            <label for='field_repeater_start'>Start ";

        gform_tooltip( names: 'form_field_repeater_start');

        echo " </label>
            <input type='number' id='field_repeater_start' min='1' value='1' onchange='SetFieldProperty('start', this.value);'>
            </li>";

        echo "<li class='repeater_settings field_setting'>
            <label for='field_repeater_min'>Min ";

        gform_tooltip( names: 'form_field_repeater_min');

        echo " </label>
            <input type='number' id='field_repeater_min' min='1' value='1' onchange='SetFieldProperty('min', this.value);'>
            </li>";

        echo "<li class='repeater_settings field_setting'>
            <label for='field_repeater_max'>Max ";

        gform_tooltip( names: 'form_field_repeater_max');

        echo " </label>
            <input type='number' id='field_repeater_max' min='1' value='1' onchange='SetFieldProperty('max', this.value);'>
            </li>";
    }
}

public static function gform_appearance_settings($position, $form_id) {
    if ($position == 408) {
        echo "<li class='repeater_settings field_setting'>
            <input type='checkbox' id='field_repeater_hideLabel' onchange='SetFieldProperty('hideLabel', this.checked);'>
            <label for='field_repeater_hideLabel' class='inline-block hide_label & Description'>

        gform_tooltip( names: 'form_field_repeater_hideLabel');

        echo " </label>
            </li>";
    }
}
```

MVC

Model, View & Controller

WHY?



Clean code

DRY Principle



Open Source

===

Giving





Reduce
Confusion

HOW?



Planning



23

Separate

23

your

23

concerns

23

Please?



Consistency



(AND OTHERS')

MY STRUCTURE for plugins

Controller

Handles setup and teardown (install, uninstall, startup, finishing tasks)

Handles hooks, actions and short codes as required

Gets data from **models** and creates **views** from that data (separate public/admin controllers in separate files)

Passes pertinent information to the **views**)

NO HTML.

Models

Creates, reads, updates and deletes (CRUD!) data

Uses WP built-in functions for handling data

Can be from custom posts, custom tables or APIs

POSITIVELY NO HTML.

Views

Made with HTML

Limit PHP code to loops and echo/print



Static files (/static/js, /static/css, /static/img)

Store all the non-processed files

Except JS can be processed, but use localization

Images, CSS, Less/Sass/that other one

ABSOLUTELY NO PHP.

File & folder structure example

- my-awesome-plugin.php
- /classes/
 - class-controller-public.php
 - class-controller-admin.php
 - class-data-model.php
- /views/
 - view-admin-page.php
 - view-public-widget.php
- /static/
 - js/
 - public.js
 - admin.js
 - css/
 - public.css
 - admin.css
 - img/
 - my-image.jpg

(AND OTHERS')

MY STRUCTURE
for themes

Controller (functions.php → /classes/class-theme.php)

Handles setup and teardown (install, uninstall, startup, finishing tasks)

Handles hooks and actions custom to your theme

Gets data from **models**

Let WordPress handle views (standard index.php, page.php, single.php, etc.)

Unless it's admin views, then you'll have to handle it

Easy access for complex data retrieval (via models) for **view(s)**

NO HTML.

Models (/classes/class-data-model.php)

Creates, reads, updates and deletes (CRUD!) data

Uses WP built-in functions for handling data

Can be from custom posts, custom tables or APIs

POSITIVELY NO HTML.

Views

Stick to WordPress' way of handling views

Less confusion for developers new to MVC development

Consistency - be consistent with other theme developers

But don't be afraid to use `/views/` for repeat use "partials" that don't fit in the WordPress structure

Static files (/static/js, /static/css, /static/img)

Store all the non-processed files

Except JS can be processed, but use localization

Images, CSS, Less/Sass/that other one

ABSOLUTELY NO PHP.

File & folder structure example

- functions.php
- index.php
- page.php
- archive.php

- /classes/
 - class-theme.php
 - class-controller-admin.php
 - class-data-model.php
- /views/
 - view-admin-page.php
 - view-reusable.php
- /static/
 - js/
 - public.js
 - admin.js
 - css/
 - public.css
 - admin.css
 - img/
 - my-image.jpg

Theoretical Scenario

Flight & Guest Cross Reference Tracker Plugin

Controllers

Main Controller (/classes/class-controller.php) - instantiates classes and handles plugin startup & pages

Callbacks (/classes/class-callbacks.php) - handle WordPress actions and filters

Models

Flight (/classes/class-flight.php) - extend the WP-HTTP class, call Flight Status API, references and adds methods to the **flight custom post type**

Guest (classes/class-guest.php) - handles data and extra methods for the **guest custom post type**

Views

single-flight.php - single flight page partial

single-guest.php - single guest page partial

guest-reference.php - repeatable guest display for flight page

flight-reference.php - repeatable flight display for guest page

MVC Plugin Boilerplate

<https://github.com/tommcfarlin/WordPress-Widget-Boilerplate>

Don't put HTML
in a function.

Keep your code
clean.

Comment your code.

Be a good open
source citizen.



Now go paint a
masterpiece.

<https://mattgraham.ca/html-in-functions/>