

The WordPress Customizer

Customizer? I barely know 'er!

Exploring the Theme Customization API

The Theme Customization API

- Since 3.4
- Create settings and controls
- Live preview
- Core features moving to the customizer

The Theme Customization API

Criticisms

- “Interface too Cramped!”
- “Customizer not mature enough!”
- “Should be for styles, not content!”

The Theme Customization API

Benefits

- Provides a central control panel in a known location
- Consistent interface for controls and settings
- Less work for developers
- “Decisions, not Options”

Panels, Sections, Settings & Controls

Oh My!

The image illustrates the WordPress Customizer interface, showing the hierarchy of settings and controls. It is divided into three main levels:

- Panel:** The top-level menu, highlighted with a red border. It includes options like "You are customizing Develop WordPress", "Site Title & Tagline", "Menus", "Lucidus", "Header Image", and "Widgets".
- Section:** A sub-menu, highlighted with a blue border. It shows settings for the selected "Menus" panel, including "Footer", "Copyright Name", "Proudly Powered By WordPress", and "Theme Information".
- Setting & Control:** The specific settings and controls for the selected section, also highlighted with a blue border. It includes a text input for "Copyright Name" (set to "Develop WordPress") and checkboxes for "Proudly Powered By WordPress" and "Theme Information".

A blue double-headed arrow labeled "Setting" and "Control" indicates the relationship between these two levels. A red line connects the "Menus" panel to the "Menus" section, and another red line connects the "Proudly Powered By WordPress" control to the "Proudly Powered By WordPress" setting.

Method: add_panel()

```
$wp_customize->add_panel('my_panel', array(
    'priority' => 10,
    'capability' => 'edit_theme_options',
    'theme_supports' => '',
    'title' => 'My Panel Title',
    'description' => 'My Panel Desc.',
));
```

Method: add_section()

```
$wp_customize->add_section('my_section', array(  
    'priority' => 10,  
    'capability' => 'edit_theme_options',  
    'theme_supports' => '',  
    'title' => 'My Section Title',  
    'description' => 'My Section Desc.',  
    'panel' => 'my_panel',  
));
```

Method: add_setting()

```
$wp_customize->add_setting('my_setting', array(  
    'type' => 'theme_mod', // or 'option'  
    'capability' => 'edit_theme_options',  
    'theme_supports' => '',  
    'default' => 'my default value',  
    'transport' => 'refresh', // or postMessage  
    'sanitize_callback' => '...', // cust->DB  
    'sanitize_js_callback' => '...', // DB->cust  
));
```


Method: add_control()

```
$wp_customize->add_control('my_setting', array(  
    'label' => 'My Custom Content',  
    'section' => 'my_section',  
    'type' => 'textarea',  
    'active_callback' => 'my_cb',  
));
```

Class WP_Customize_Control()

Default Control Types

- text (default)
- checkbox
- radio
- select
- dropdown-pages
- textarea

Class WP_Customize_Control()

Additional Control Types

- WP_Customize_Color_Control()
- WP_Customize_Upload_Control()
- WP_Customize_Image_Control()

Class WP_Customize_Control()

Create a Custom Control Class

```
class My_Custom_Control extends  
WP_Customize_Control {  
  
    public function render_content() {  
        echo $my_custom_markup;  
    }  
}
```

Class WP_Customize_Control()

Create the Custom Control

```
$wp_customize->add_control(new My_Custom_Control(
    $wp_customize,
    'my_setting',
    array(
        'label' => 'My Custom Control',
        'section' => 'my_section',
        'choices' => array(
            'one' => 'choice one',
            'two' => 'choice two'
        )
    )
));
```

Using the Settings in Your Theme

```
function: get_theme_mod()
```

```
get_theme_mod( 'my_setting', 'My Default' )
```

Action: wp_head

```
add_action( 'wp_head', 'my_custom_styles' );
```

```
function my_custom_styles() {  
    printf( '<style>#my-el{ color: %s; }</style>',  
        get_theme_mod( 'my_setting', '#fff' ) );  
}
```


Using Settings Inside Your Template

```
<div id='my-el'>
```

```
<?php echo get_theme_mod( 'my_custom_content',  
'My Default Value' ); ?>
```

```
</div>
```

JavaScript

Live Styles

- Connects settings to the preview window
- Applies styles without a page refresh
- Use in conjunction with 'transport' => 'postMessage'

Action: 'customize_preview_init'

```
add_action( 'customize_preview_init',  
    'enqueue_my_live_styles' );  
  
function enqueue_my_live_styles() {  
    wp_enqueue_script(  
        'my-live-styles',  
        get_template_directory_uri().'/my-live-styles.js',  
        array( 'jquery', 'customize-preview' ),  
        '',  
        true  
    );  
}
```

my-live-styles.js

```
( function($) {  
    wp.customize( 'my_setting', function( value ) {  
        value.bind( function( to ) {  
            $( '#my-el' ).css( 'background', to );  
        });  
    });  
});  
}
```

Coming Soon: Menu Management

Feature Plugin: Menu Customizer

- Adds menu management controls to the customizer
- Introduces a new transport method: partial refresh
- Will soon be part of core

Customizer Resources

- <https://developer.wordpress.org/themes/advanced-topics/customizer-api/>
- https://codex.wordpress.org/Theme_Customization_API
- https://developer.wordpress.org/reference/classes/wp_customize_manager/
- <https://make.wordpress.org/core/2014/07/08/customizer-improvements-in-4-0/>